Optimum Trajectory Planning for Redundant Manipulators Through Inverse Dynamics

Ayten, Kagan

Award date:
2012

Awarding institution:
University of Bath

Optimum Trajectory Planning for Redundant Manipulators Through Inverse Dynamics

submitted by

Kagan Koray Ayten

for the degree of Doctor of Philosophy

of the

University of Bath

Department of Mechanical Engineering

December 2012

CONTENTS

ABSTRACT

The purpose of this thesis is to develop methods to generate minimum-energy consumption trajectories for a point-to-point motion under pre-defined kinematic and dynamic constraints for robotic manipulators. With respect to other trajectory optimization methods, the work presented in this thesis provides two new methods to the scientific literature. The proposed methods improve the handling of the constraints in trajectory optimization methods as well as reducing the computational complexity of redundant/hyper-redundant manipulator systems.

The proposed constraint handling method has the advantage that the system's kinematic and dynamic constraints are handled in a sequential manner within the cost function to prevent running the inverse dynamic solution when the constraints are not satisfied during the desired task. Thus, the complexity and computational effort of the optimization algorithm is significantly reduced. This proposed method is also applicable to the other types of robotic manipulator such as redundant, hyper redundant and parallel robots. A number of theoretical and experimental simulations are performed and also three different types of robotic manipulator concepts such as 2-links non-redundant, 3-links redundant and 8-links hyper-redundant manipulators have been considered to demonstrate the effectiveness of the proposed method.

In addition to that, an efficient control algorithm was investigated for redundant robotic manipulators. The redundancy offers significant benefits in increasing the robotic manipulator's flexibility and versatility. When a robotic manipulator has re-

dundancy, it has the ability of moving each joint in infinite ways for the same specified end-effectors position. In order take full advantage of the redundancy, an efficient control algorithm is needed. The basis of the control algorithm developed is to utilise a virtual link concept which replaces all the redundant links to eliminate physically impossible configurations before running the inverse dynamic model. In this case, all of the redundant links are considered as a single link and control complexity of the redundant/hyper redundant links is significantly reduced. Control forces are also calculated for each link. An 8-link hyper-redundant manipulator is also studied. The validity of the proposed virtual link approach is supported by simulation studies.

Although feasible optimum solutions have been provided by the trajectory optimization process for the desired trajectories, the global minimum point is not evident or clearly met for all of these simulation results. On the other hand, significant reduction in the cost values was provided as a result of the proposed trajectory optimization method, which was initially performed using computer simulations and then same intended trajectories have been implemented on the Katana 450 robotic manipulator. After optimising the desired task for the non-redundant study, the cost values are reduced significantly. The greatest reduction in the energy consumption made was 55.8 % in 8 seconds amongst the other motion durations of the theoretical studies and the corresponding experimental energy saving rate is 43.37 % for the intended task for 8 seconds. In addition to this, the cost values are also reduced remarkably in the redundant study, and the considerable reduction in the cost made was approximately 56.57 % and 43.001 % in 4 seconds amongst the other motion durations, for the theoretical and experimental studies, respectively. In a more complicated theoretical hyper-redundant study, energy reduction was significantly promising, and the cost value was reduced from $G = 8084$ to $G = 5232$, which corresponds to 35.3 % reduction along the desired trajectory. Thus, these results confirm the effectiveness of the proposed methods.

Finally, an error analysis of the robotic manipulators along the trajectories is also presented in order to investigate tracking accuracy for the desired trajectories with different motion durations.

ACKNOWLEDGMENTS

A PhD thesis could not be completed by the work of one person; a variety of people provided priceless support and encouragement during the procedure of research and writing.

I would like to tremendously thank Dr. M. Necip SAHINKAYA for providing me this excellent, once in a life time opportunity to work within the Mechanical Engineering at the University of Bath. In fact, his professional recommendations have permanently been of great support, which pointed me in the right direction from the very beginning of my PhD study. His inspirational words have assisted to stimulate me completely and lift me up when the times were difficult. His generous support and attention to details have been essential in making this thesis possible. This PhD would have not been accomplished without his priceless support and mentoring. He has taken time out of his very busy schedule to give me significant feedback on my work and PhD thesis. He welcomed me with endless, tireless support and provided invaluable assistance in my project work. It has been a great pleasure to work with supervisors of such high standard.

Also Dr. P. IRAVANI's guidance, support and suggestions are much appreciated.

To my parents: I am ever grateful for your unflinching supports. In particular, thanks to my father Huseyin AYTEN and my mother Hatice AYTEN for their indefinable support, amazing love and continuous encouragement at all times under all circumstances. They were with me all the time and they always made me and my sister

very happy. They gave me the extra strength and motivation essential to get things done. Their incredible positive attitudes, personalities and encouragements have not gone unnoticed. I am so grateful and also so lucky because you are my family.

I would like to express my gratitude to my wife Nurgul AYTEN and my sister Nilay Besen AYTEN for their cooperation, love and support. My beautiful daughter Hatice Selin AYTEN and my handsome nephew Cagan Tuna AYGUN, who make me always feel very happy and hopeful. Thanks for this.

To my grandfathers Sadik AYTEN, Bilal UZUN and my grandmothers Elmas AYTEN, Habibe UZUN, I am so proud of you and you will always be in my heart.

Thanks go to Orhan UZUN for his unflinching and well-intentioned support, even if he is in difficult and desperate situations.

Suleyman YALDIZ, Edip Ali YALDIZ and their family's support and opinion have been of great help during my PhD study and it is unforgettable.

Yuksel DAL's friendship and financial assistance will not be forgotten.

To Fehmi BODUR: He has an amazing and wonderful personality. I will never forget his support and friendship.

I am also thankful to Dr. Suat BEKIRCAN for his comments on my thesis.

Finally, special thanks to Murat B. ARSLAN, Huseyin KOCAK and Mehmet MENCOGLU. Your support and friendship will never be forgotten.

NOMENCLATURE

Variables

$\mathbf{0}$ = Matrix of zeros

$\mathbf{A}$ = ZxZ matrix of $a_{i,j}$ functions

$\mathbf{B}$ = Control action specifying coefficient matrix

$\mathbf{C}_1$ = Matrix of functions $\mathbf{q}$ and t

$\mathbf{C}_2$ = Matrix of function $\mathbf{q}$, $\dot{\mathbf{q}}$ and t

$\mathbf{D}_1$, $\mathbf{D}_2$ = Vectors of function $\mathbf{q}$, $\dot{\mathbf{q}}$ and t

$\mathbf{F}$ = Constraint Jacobian Matrix

$\mathbf{Q}$ = Vector of generalised inputs

$\mathbf{q}$ = Vector of generalised coordinates

$\mathbf{U}$ = Control input vector

$\mathbf{y}$ = Desired motion specifying coordinates

λ_j = Lagrange multiplier

θ_{deg} = Angular position of linkage n

θ_{enc} = Encoder position of linkage n

$\theta_{red_{non_{opt}}}$= Non-optimum B-spline parameters in joint space for redundant case

$\theta_{red_{opt}}$= Optimum B-spline parameters in joint space for redundant case

$Acce_{max}$ = Maximum permissible output acceleration

$a_{i,j}$ = Function of $\mathbf{q}$ and time

bb = Large base values for alternative cost function

$c(n)$ = Nonlinear inequalities

e = Error value

f_j = Constraint equation

G = Alternative cost value

g_i = Required actuator torques to be applied at join i

i, j, k = Integers

$Joint_{\theta_1}$= Non-optimum B-spline parameters in joint space for θ_1 (non-redundant case)

$Joint_{\theta_2}$= Non-optimum B-spline parameters in joint space for θ_2 (non-redundant case)

K = Number of degrees of freedom of the required motion

L = Lagrangian function

l_n = Length of linkage n

M = Number of degrees of freedom

$max(\theta_{enc})$ = Maximum encoder position of linkage n

m_{load} = Mass of load n

m_n = Mass of linkage n

n = Linkage number reference

$n_{section}$ = Fifth order B-spline section

$P_{initial}$ = Initial position of the motion

P_{final} = Final position of the motion

p = The number of cost function calls

Q_i = Generalised output

q_i = Generalised coordinate

R_{eef} = Distance between the origin and end-effector of the manipulator

R_n = Gear ratio

R_{rm} = Distance between the origin and end point of the redundant links

R_{nrm} = Distance between end point of the end-effector and last point of the redundant link

r_n = B-spline control points

$S_i(t)$ = A third degree cubic polynomial at a given time, t

s = Output current

T = Duration of motion

t = Time

t_{enc} = Time in encoder unit

V_{max} = Maximum capable velocity

x_{eef}= Non-optimum B-spline parameters in Cartesian space for x direction

x_f= End-effector final position in x coordinate

x_i= End-effector initial position in x coordinate

$x_{non_{opt}}$= Non-optimum B-spline parameters in Cartesian space for x direction for redundant case

x_{opt}= Optimum B-spline parameters in Cartesian space for x direction for non-redundant and redundant cases

y_{eef}= Non-optimum B-spline parameters in Cartesian space for y direction

y_f= End-effector final position in y coordinate

y_i= End-effector initial position in y coordinate

$y_{non_{opt}}$= Non-optimum B-spline parameters in Cartesian space for y direction for redundant case

y_{opt}= Optimum B-spline parameters in Cartesian space for y direction for non-redundant and redundant cases

Z = Number of generalised coordinates

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

In today's modern industrial world, robotic manipulator's aim involves developing the quality of the products, improving the productivity, accuracy, speed, and flexibility. The requirements of industrial applications are vastly complex and difficult, but despite this, they can be employed in hostile environments and perform tasks that humans do not want to do (e.g., dangerous, boring or continuous repetitive tasks). Seeking a more efficient manufacture and better quality work in industry is an essential industrial consideration, which led to the improvement of modern and dexterous robotic manipulator systems. However, these skilful robotic manipulators are hardly autonomous and require initial actions (such as calibration, manipulator trajectory planning) in order to meet demands on certain tasks. Therefore, it is extremely attractive to control their action properly in the working environment to successfully execute the given task. The capability of executing precise missions and to track a required motion trajectory lies at the heart of the robotics research [3]. The major research areas of the robotic field can be summarized as follows:

- Robotic manipulator design
- Artificial intelligence
- Trajectory planning

Among these scientific research fields, the issue of trajectory planning has a crucial place in the robotic field. When the robot is to be moved from initial position to final, path and trajectory planning problems are addressed and taken into account by numerous researchers. Trajectory planning problems are considered as a very active research area and accepted as two distinct parts of the robotic field. There is a clear difference between the path and trajectory planning for robotic systems. The path planning considers obtaining a path of robot configurations between an initial and a final position that provide collision free trajectory. Whereas, optimum trajectory planning can be identified as the procedure of choosing a movement and the associated optimal controls from the set of admissible movements and controls, while satisfying all the kinematic and dynamic constraints and minimizing a required cost function such as; the required time of the energy consumption [4]. Therefore, path planning problems are a subset of trajectory planning issues. Optimum trajectory planning is extremely vital to the efficient operation of a particular application.

In many industrial automation tasks, large numbers of robotic manipulators are in utilised in the production line. These robots perform tasks such as welding, pick and place operations and many other tasks. The robotic manipulator industry has grown rapidly and this improves the productivity as well as the number of robotic manipulators in use every year. Most of the industrial robotic manipulators are driven electrically and execute their jobs continuously every day. In the industrial sector, the industrial applications have repetitive processes for an intended job and a large amount of energy is demanded during the given application [5]. If a robotic plant has multiple installation lines or multiple facilities, total energy consumption will be significantly increased during the given processes.

In order to increase the productivity in required processes, minimum execution time is widely utilised as a trajectory optimization criteria in order to decrease the production time, and to maximize the profitability [6]. Although the minimum execution time factor is an inherent trajectory optimization criterion and an important industrial consideration, it is not convenient if a smooth trajectory for the robotic motion is required. This kind of robotic trajectory can result in undesirable shocks and vibra-

tions, increased energy consumption and loss of accuracy for the given path. Also, all manipulator's actuators operate at their maximum output. This has an impact on the robot's gears, actuators and the manipulator structure [6]. Whereas, generating the robot trajectory by taking into account the energetic criteria provides some benefits for the intended applications. It can provide smooth tracking performance, less stress on the robotic actuators and does not fluctuate as much as in the time optimal motion.

The work detail in this thesis explores methods for developing efficient constraint handling method for non-redundant and redundant robotic manipulators on the basis of minimizing energy consumption. The actuator torque has been considered for the formulation of the cost function for the theoretical study, but the current of each motor was utilised for the experimental study using a Katana 450 robotic manipulator. The method developed will take into account both mechanism kinematics and dynamics constraint conditions.

1.1 Research Topics of the Thesis

This thesis focuses on optimal trajectory planning for non-redundant and redundant/hyper-redundant robotic manipulators. This research field proposes a trajectory planning method and it aims to generate an optimum trajectory based on minimum torque and/or energy consumption for the industrial robotic manipulators.

Most of the industrial robotic manipulators consist of six or less degrees-of-freedom (DOF) and execute basic and continuous repetitive jobs such as electronic assembly, welding and automated packaging. There are an infinite number of movements a robotic

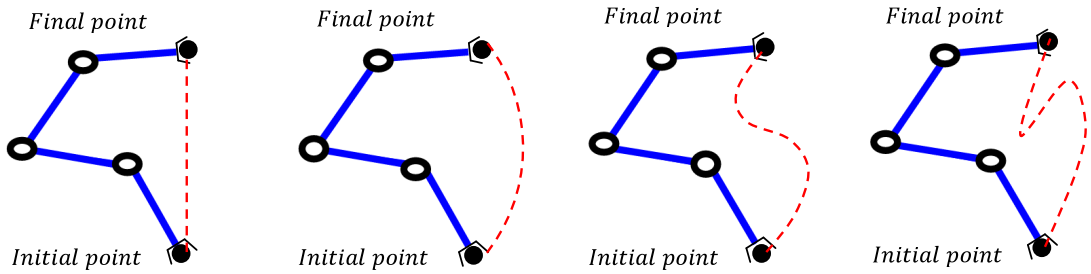

Figure 1-1: *Path variations of the same desired trajectories between initial and final point.*

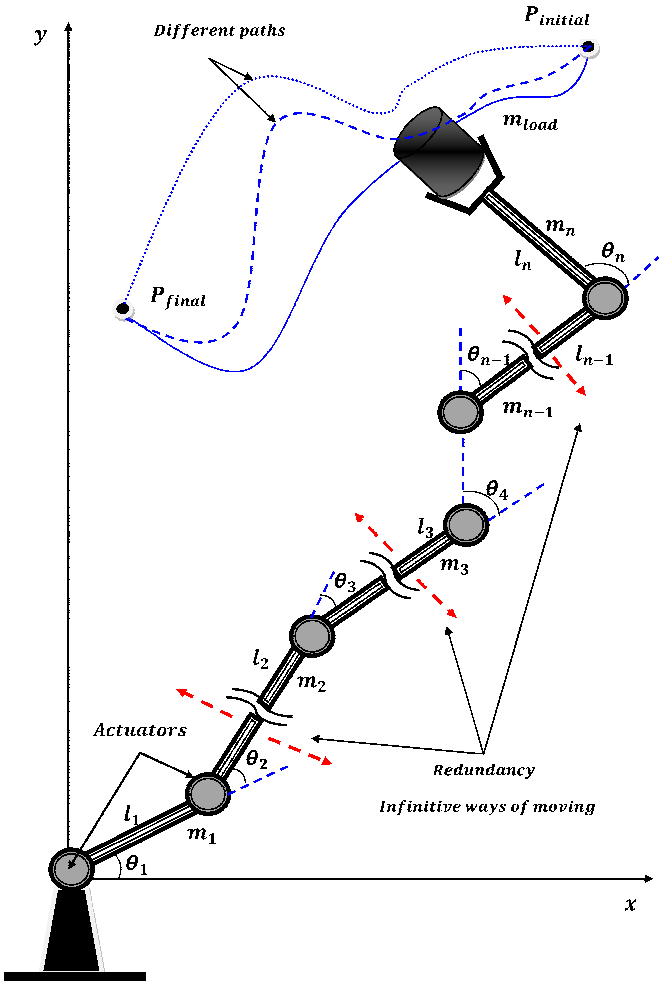

Figure 1-2: *Schematic view of a redundant manipulator.*

manipulator can perform to move between an initial and a terminal state as in Fig. 1-1 during the desired application. Each trajectory of the robotic manipulator will consume a varied amount of power and the energy utilised by a manipulator's actuator is related to actuator torques. Therefore, the trajectory planning problem is taken into account to be an optimization problem optimizing some parameters to minimize a required cost function subject to a number of kinematic and dynamic constraints.

However, there are also other complicated applications which demand better performance and flexibility (drilling, cutting, medical robotics, maintenance of nuclear reactors etc.). In order to meet these demands, robotic manipulators may have more degrees-of-freedom than essential to execute intended complicated jobs like human arms [7]. That is, a redundant robotic manipulator has additional joint parameters

that allow the manipulator to change its position or orientation when the manipulator's end-effector is fixed. For instance, the human arm consists of 7 DOF, three at the shoulder, one elbow joint and three at the wrist. In this case, humans can easily move their elbow around a circle, when the wrist is in a fixed position. Hence, this provides us the ability to move our arm more freely. The extra degrees-of-freedom can be named redundant, and this also improves the robotic manipulator's abilities. Redundant robotic manipulators have many advantages over non-redundant ones and can also be utilized effectively to handle other optimization criteria such as obstacle avoidance in working space, avoiding singularities, avoiding limits of joints, minimizing torque/energy (optimization of some cost function) over a given job and also ensures a high degree of manipulability during the execution of the required motion of the end effectors [7]. The redundancy can be defined as;

"*When a manipulator can reach a specified position with more than one configuration of the linkages, the manipulator is said to be redundant.*" [7].

Generally, inverse kinematic solutions of non-redundant robotic manipulators present minimal numerical computational complexity. When a robotic manipulator has redundancy, it has the ability of moving each redundant joint in infinite ways for the same specified end-effector motion as shown in Fig. 1-2. Therefore, despite their important features, inverse kinematic solutions and control of redundant/hyper-redundant robotic manipulators are more complicated and the trajectory planning issue also becomes increasingly hard with each added redundant degree-of-freedom. In this case, the inverse dynamic equations consist of more unknowns than the number of equations. The redundant control inputs can either be arbitrarily defined or additional constraint equations may be supplied [8].

1.2 Motivation

In order to solve the trajectory optimization problems, various trajectory optimization algorithms have been developed and implemented by numerous researchers. All these developed trajectory optimization procedures have their own individual strengths,

weaknesses and features [3].

Ideally, in order to determine the optimum solution successfully, computationally efficient optimization procedures are preferred for a given task. This is because, utilised trajectory optimization algorithms can be a numerically and computationally extremely complex and challenging issue due to the large number of optimization parameters and various constraints which need to be handled effectively during the optimization process [9].

The optimization process consists of the cost function calculations which involve running the computationally intensive inverse dynamic model. If the created inverse dynamic model is redundant or hyper-redundant manipulators, inverse kinematic solutions and control of redundant/hyper-redundant manipulators become more and more complicated and the trajectory planning problem also becomes increasingly difficult with each added redundant degree-of-freedom [10]. In this case, the optimization procedure will be a time consuming problem for a given trajectory in order to determine the feasible optimum solutions.

For the success of any optimization procedure, the trajectory optimization algorithm should be easily used on various types of machines such as non-redundant, redundant and hyper-redundant manipulators and the various types of constraint equations should be handled effectively during the trajectory optimization procedure.

Therefore, handling the constraints effectively and reducing the control complexity of redundant manipulators are a crucial motivation factor and also interesting challenge for this PhD thesis.

1.3 Novelty of Work

The novelty of work claimed in this thesis includes:

- The optimization algorithm is computationally efficient as kinematic and dynamic constraints are included in the cost function to prevent running the inverse dynamic model when all constraints are not satisfied (Chapter 4).

- The optimization algorithm and also proposed alternative cost handling method can be applied to all types of robots such as non-redundant, redundant or hyper-redundant robots in order to optimize the desired trajectory (Chapter 5).

- The novelty of this redundant control scheme relies on the definition of a virtual link concept, where all the redundant links are acting as a single link. Hence, it makes controlling these links easier and control complexity of the redundant/hyper-redundant manipulators is reduced. This process is applicable to hyper-redundant manipulators with a large number of links (Chapter 6).

- A novel virtual link concept allows us to replace all the redundant links to eliminate physically impossible configurations before running the inverse dynamic model. Therefore, this control algorithm prevents inverse dynamic failure (even if the manipulator is within the workspace) during the optimization process (Chapter 6).

The following benefits were also considered in this thesis:

- Desired trajectory is based on continuous functions.

- The optimization procedure considers dynamic cost functions through an inverse dynamic analysis and hence control forces are also calculated for each link of redundant manipulators (Chapter 6).

1.4 Aims and Objectives of Research

This thesis contributes to the field of trajectory optimization for non-redundant, redundant and hyper-redundant robotic manipulators.

The aim of this work is:

- To develop techniques to generate minimum energy consumption trajectories for non-redundant, redundant and hyper-redundant robotic manipulators under kinematic and dynamic constraint conditions.

The objectives of the project were:

- To develop a generic method to calculate optimum trajectories to minimize energy consumption in fully actuated robotic manipulators under dynamic and kinematic constraints. Energy efficiency will be related to the magnitude of required joint torques during the desired motion.
- To develop a generic technique to calculate optimum trajectories and control inputs for redundant manipulators to minimize energy consumption. The technique should handle singularities, joint limits, obstacle avoidance, and a large number of links.

In both objectives, a theoretical simulation study will first be performed to develop and examine the effectiveness of potential techniques. These techniques will then be verified experimentally by using industrial Katana 450 robotic manipulators.

1.5 Layout of the Thesis

In this thesis the following structure is used:

Chapter 1 provides an introduction to non-redundant/redundant robotic manipulators and problems of the trajectory optimization techniques. Also, the aims and objectives of research and novelty of work are given.

Chapter 2 gives a literature review of trajectory optimization methods covering both non-redundant and redundant/hyper-redundant manipulators.

The experimental setup and modelling of the test rig are detailed in Chapter 3. In this chapter, an off-line trajectory planning method will also be presented for non-redundant/redundant robotic manipulators.

Chapter 4 details work on the procedure behind the optimization methods and some background is given in order to achieve minimum energy consumption trajectories for a point-to-point motion under kinematic and dynamic constraints. To derive the trajectory, a multi-parametric optimization method is used. A proposed penalty algorithm and proposed trajectory optimization procedure are also given.

Chapter 5 presents theoretical and experimental results on non-redundant optimization systems in order to validate the proposed penalty algorithm described in Chapter 4. A two-link Katana robotic manipulator is used in 2D.

Chapter 6 focuses theoretical and experimental results on redundant manipulator systems and also trajectory optimization strategies to minimize consumed energy. A virtual link concept will be introduced for redundant links in order to reduce the control complexity of redundant/hyper-redundant manipulators. A three link Katana manipulator will be used based on links 2, 3 and 4 in order to validate the virtual link concept. Also, an 8 link hyper-redundant manipulator is theoretically simulated to demonstrate the applicability of the proposed virtual link concept to the hyper-redundant manipulators.

Finally, Chapter 7 includes a summary of conclusions of the work detailed in this thesis. Recommendations for future work are also included.

CHAPTER 2

LITERATURE REVIEW

In the scientific literature, different types of trajectory optimization procedures have been developed in order to speed up the trajectory optimization procedure, however, this procedure is sometimes accomplished at the expense of absolute precision and also jammed in local minima point as a result of the optimization. Other trajectory optimization techniques have also been addressed in order to get very accurate and precise numerical optimum solutions, however, at the expense of computational speed for the optimization procedure.

This chapter presents an analysis of the literature concerned with various optimum trajectory planning procedures for a variety of types of robotic manipulators. Optimality criterions for non-redundant, redundant and hyper-redundant manipulator strategies are reviewed in sections 2.1, 2.2 and 2.3 respectively.

2.1 Trajectory Optimization for Non-Redundant Manipulators

In order to implement the desired task, any manipulation of the robotic operation can be executed in numerous ways. However, a most convenient option can be shown as the robotic manipulator carries out the essential task in an optimal manner with respect to

some relevant optimality criteria. These criteria cover minimum time, minimum jerk, minimum energy consumption and also multi-criteria by handling the manipulator's kinematic and dynamic constraints effectively.

In this section of the chapter, various optimization techniques and also optimization criteria of the non-redundant robotic manipulators are given and discussed.

2.1.1 Time Optimality Optimization

In numerous industrial implementations in the late 1960s, the speed of the robotic manipulators in the industry was too slow and hereby their productivity quite depended on the ability and capability of their actuators. The best solution of increasing the productivity would be increasing the actuator capacity and power for a given task. However, the cost and power consumption for each production line as well as the inertia of the overall robotic manipulator's system would be increased inherently by utilising the larger size of robotic actuators. Therefore, in order to increase productivity in the industrial demands, the research of minimum-time optimal algorithm can be addressed and it always relates to maximizing efficiency of robotic manipulator's systems. There are numerous engineering researches and implementations where minimum-time optimal control is desired for a given task.

Research on the optimal minimum-time approach for robotic manipulators dates back to the early 1970's. Kahn and Roth [11] are amongst the first researchers to deal with the minimum-time optimal control issue of robotic manipulators for point-to-point motions in the given trajectory. In their research, a three-link serial robotic mechanism was taken into account, and restrictions on the magnitude of the actuator torques were supposed to be constant during the desired motion. To move the end-effector of the robotic manipulators along the path in optimum minimum time criterion under the given torque constraints, the maximum allowable actuator torques were determined by an optimization algorithm. Trajectory of the robotic manipulator was not specified, but only the final destination of the robotic trajectory was given. Although this work was convenient for some robotic tasks, many robotic implementations often require essential trajectory planning to specify the robotic manipulator's trajectory in order to

prevent collision with obstacles.

To solve the minimum-time optimal problem with collision avoidance, Nix and Auslander [12] utilised a parameter optimization scheme by taking into account the switching points at which actuators of the manipulator switch from the maximum torque magnitude to minimum or vice-versa. In their proposed optimization technique, the maximum control torque (bang-bang control) is applied by each joint actuator in the system while allowing robotic manipulator to prevent collision with all obstacles and achieve its end point [12]. However, the proposed optimization method may not be easy to implement for most real applications in the industry due to the extensive computational time.

Another optimal minimum-time approach with collision avoidance for robotic trajectory was described by Dubowsky and Shiller [13]. A six degrees-of-freedom robotic manipulator system was executed experimentally for along any given path to demonstrate the optimal minimum-time trajectory. In their proposed optimization algorithm, in order to provide the shortest execution time and determine the one that prevents the collision with all obstacles, the desired trajectory was varied for the desired task.

In addition to above works, another practically feasible optimal minimum-time trajectory optimization technique for along the specified trajectory was performed by Bobrow et al. [14], who presents in detail the solution derived by Bobrow [15] and Bobrow, Dubowsky and Gibson [16] and Shin and McKay [17]. In his proposed method, the robotic manipulator's trajectory was parameterized with a single variable of position profile. Hence, the multi-dimensional trajectory optimization issue was minimized to a single dimensional one. A velocity limit curve was generated in the position velocity plane by taking into account the constraints on the manipulator's actuator. These constraints represented the feasible trajectory in the position-velocity plane for the optimal minimum-time trajectory for a given task. Hereafter, the optimal minimum-time trajectory was then achieved from these given limits. In this case, the trajectory optimization algorithm utilised only one single iteration variable to determine the switching curve in the position-velocity plane.

The basic concept of their optimization methods was the selection of an acceler-

ation profile by the optimum minimum-time solution in order to deliver the largest velocity such that, at every point on the trajectory, the speed of the manipulator must not exceed the maximum speed profile of the system to maintain the end-effector's position on the desired trajectory. The optimum minimum time trajectory solution was determined by obtaining the switching curve point in the position-velocity plane (phase plane) along the given trajectory. The idea of the switching curve point is a graphical representation and a good indicator to show the optimum minimum-time solution [14], [17]. That means that, if the velocity profile of the actuator stays under the switching curve points, an optimal solution was defined by the maximum acceleration profile of the actuator. On the other hand, when the velocity profile of the actuator stays on the switching curve points, optimal trajectory solution was defined by either maximum acceleration profile or maximum deceleration profile of the actuator, depending on the position profile on the switching curve [14], [17].

The full non-linear robotic manipulator dynamics and the torque constraints have been taken into account by all of the above researchers, thus, the proposed optimization methods in these studies provide the optimal minimum-time solution. The study of [14], [17] were followed by Pfeiffer and Johanni [18]. In their proposed method, they expressed the parameterized dynamic equations that differ from the earlier researches while utilising fundamentally the same procedure as in Shin and McKay [17]. In Pfeiffer and Johanni's work, a switching point was described by sinks and sources along the maximum velocity curve profile for the proposed manipulator's trajectory. A switching point can take place at tangency points where one of the manipulator's actuators is tangent to its saturation limits [19] or at either critical point where the acceleration profile at the limit curve is not unique [18]. At a tangential point, the allowable acceleration profile is unique, and the robotic manipulator's trajectory is inherently tangent to the limit curve, where at a critical point, the acceleration profile can be chosen from a limited feasible range. However, at some crucial points where the robotic manipulator is being forced to pass the limit curve due to the maximum acceleration profile, the assumption of maximum acceleration profile or deceleration profile may be ineffective [19]. On the other hand, if the robotic manipulator's trajectory does not have

permission to cross the limit curve intentionally, the mentioned trajectory optimization algorithms in the above get trapped at such points during the desired motion. The violation can be found with at least one of the robotic actuator's limitations due to crossing the limit curve during the desired motion; as a result of this a deviation may occur [19]. In addition to this work, additional limitations to the velocity limit curve profile were added by Slotine and Yang [19] in order to make the original optimum trajectory planning algorithm more efficient in their proposed research.

The aforementioned methods [14], [17] have been improved and extended by Shiller and Lu [20] in order to handle the case where the assumption of maximum acceleration and maximum deceleration profiles along the solution curve profile is no longer valid, and this method fails to deliver the correct answer under some circumstances [20].

All of the research mentioned above was based on two fundamental approaches. The first approach takes into account the position-velocity plane. In this approach, the manipulator actuator constraints (without jerk profile) were taken into account in order to identify the boundary curve (velocity limit curve profile) for the system. Utilising the parametric boundary curve profile (velocity limit curve profile), the switching points in the system were determined and a bang-bang type of trajectory was made and generated by taking into account the maximum acceleration and maximum deceleration sections. On the other hand, the optimal control structure for the desired robotic manipulator's trajectory was utilised in the second approach and this optimal control problem has been performed with a number of varied approaches depending on the desired control objective in the given task.

If the desired robotic manipulator's task consists of a pre-defined trajectory such as laser cutting, arc welding, etc. for the optimal trajectory planning for non-redundant robotic manipulator, this is the case investigated by Zlajpah [21]. In his proposed research, an optimum minimum-time movement over a pre-defined manipulator's trajectory was taken into account subject to the dynamic and kinematic system constraints. Zlajpah also followed the approach (parameter phase plane) of Bobrow [14] and Shin and McKay [17], nevertheless his proposed optimal-time algorithm was different on the basis of several aspects. In his research, other constrainable factors like joint velocity

profiles and task requirements (path velocity and path acceleration) were also taken into account.

However, the robotic manipulator's trajectories were generated with discontinuous values of accelerations and joint torque profiles in the aforementioned optimum minimum time algorithms [11], [12], [13], [17], [18], [19], [20]. This is because the dynamic model of the manipulator was assumed to be completely rigid for the optimal trajectory calculation, therefore, the actuator dynamics was not taken into account in the created system.

This assumption leads to an undesirable effect, for instance, the discontinuous torque profiles cannot be created by the actual actuators of the robotic manipulator due to their physical limitations [22], [23], [24]. This discontinuous torque profile always leads to a delay in the joint motion with respect to the desired manipulator's trajectory, hence, this delay causes a reduction in the tracking accuracy of the robotic manipulator as well as undesired vibrations in the desired system. In order to overcome such problems, another time-optimal control strategy is performed by Constantinescu [25]. In his proposed research, smooth and optimal minimum time path constrained trajectories for industrial robotic manipulators were presented by taking into account the constraints on the actuator torques, torque rates and dynamics of the robotic manipulator. The cubic spline functions were utilised to define the manipulator's trajectory, which was in the position-velocity plane. The desired smoothness of the robotic manipulator's trajectory was achieved by imposing constraints on the joint torque profile. The jerk value was selected as the controlled input for the desired system. In this way, although the created optimal-trajectory was not exactly time-optimal, it was close enough to the optimality rate. Experimental studies indicated that the tracking accuracy was improved via trajectory smoothness.

Some recent research on optimum minimum-time trajectory planning for non-redundant robotic manipulators was presented by Bianco and Piazzi [26], which is an extension of their previous work [27]. The cubic spline functions were utilised to define the trajectory of the manipulator, and the continuity of the first and second derivatives of the position profile were guaranteed. The limits on joint torque profile and torque derivatives

were also taken into account for the non-linear robotic manipulator dynamics. The cost function of the optimization algorithm was defined by the total execution time and a hybrid genetic/interval trajectory optimization algorithm was utilised in order to solve a resulting global minimum-time trajectory problem [28]. A two-link planar non-redundant robotic manipulator and PUMA six-links robotic manipulators were utilised by the trajectory optimization algorithm. Comparisons with an alternative optimization solver (the MatLab optimization tool) were exposed, and the advantages of the genetic/interval optimization algorithm were highlighted.

In a more recent research, Valero [29] utilised an optimization algorithm which was capable of obtaining feasible manipulator configurations between the initial and the final positions of robotic manipulators in order to develop a trajectory for industrial robotic manipulators in workspaces with obstacles. His proposed work consisted of two successive sections, a discrete configuration space, and an optimal and feasible robotic manipulator trajectory [29]. In the first section, the two feasible robotic manipulator configurations were given to compose a discrete configuration space was composed of. In the second section, the selected configuration is used to determine a minimum weighted free-collision trajectory. Thus, feasible manipulator trajectories were achieved in joint space configuration that minimize the total execution time and were dynamically compatible with the robotic manipulator's features [29].

Another optimal minimum time robotic manipulator trajectory with collision free motion for industrial robots in a complex environment was introduced by Rubio [30]. This research was an extension of previous research done in [29]. An optimal minimum-time criterion and the dynamic of the robotic manipulator were taken into account for a trajectory optimization algorithm subject to the real working constraints such as torque, power, jerk, energy consumed, and collision avoidance. The trajectory optimization algorithm was executed on a discrete configuration space by utilising the inverse dynamic scheme [8] for the optimization technique. Obstacles avoidance is usually addressed during the path planning, and it will be discussed in section 2.2.2.

For any trajectory optimization algorithm, looking for all possible optimization parameters for the optimal robotic trajectories are a crucial and complicated issue and

lies at the heart of all trajectory optimization process. The selection of a trajectory optimisation technique is a crucial problem to determine the optimal optimization parameters for the desired robotic manipulator trajectory. Different types of optimization techniques have different features. For instance, in order to identify the global minimum point [31], [32], some trajectory optimization techniques have better strength to avoid local minima point during the desired task, and some trajectory optimization algorithms are better for local minimum search with computational efficiency. On the other hand, most of the trajectory optimization algorithm can be undesirably attracted by local minima point during the intended task [33].

In order to overcome local minima issues for the minimum-time optimization criterion, a harmony search (HS) algorithm was utilised by Tangpattanakul and Pramin [34] for optimum minimum-time trajectory planning of robotic manipulators. A minimum interval execution time of the robotic manipulator was taken into account as the objective function subject to the kinematic constraint profiles (such as velocity, acceleration and jerk). A set of cubic spline functions was utilised to define the trajectory. In their proposed study, an advantage of the HS over the Sequential Quadratic Programming technique (SQP) was clearly explained [34]. Defining the initial optimization parameters are not essential for the HS, while the SQP algorithm needs to define the logical initial optimization parameters before initiating the optimization process in order to prevent ending up in a local minimum point. Therefore, the logical initial optimization parameters finding process was avoided by utilising the HS algorithm.

In this section of the chapter, minimum-time algorithms for the trajectory planning were taken into account. The next section of the chapter will discuss the minimum-jerk criterion for the desired trajectories in order to reduce the vibration during the desired motion.

2.1.2 Jerk Optimality Optimization

Smooth motion profiles can be achieved for robotic manipulator's trajectories by minimizing the joint jerks profile, which is the derivative of the acceleration profile. The positive outcomes [35] of taking into account optimal jerk criterion can be given as

follow:

- The path deviation of the robotic manipulator can be reduced by introducing optimal jerk criterion, therefore, a highly coordinated and inherent robotic trajectory motion can be observed.
- Optimal jerk criterion results in smoothed robotic manipulator's actuator loads [36], thus, the stresses on the robotic actuators and the structure of the robotic manipulator will be reduced effectively so that the life span of the actuators will be expanded.

In the robotics literature, several optimal jerk criteria are taken into account for the trajectory optimization algorithm. The jerk-limited manipulator trajectory generation for a point-to-point motion profile was considered in some previous research [36]. The time integral of the squared jerk was taken into account in order to build a cost function with the free execution time.

In another study [37], a maximum execution time constraint was considered for a non-redundant robotic manipulator for a prespecified Cartesian space trajectory for a given task.

Simon and Isik [38] also introduced optimal jerk criterion for the optimization method. The interpolation was performed through fourth-order trigonometric spline functions to provide continuity of the jerk profile. Thus, continuity of the first three derivatives was ensured in the desired trajectory.

The minimization of jerk profile under the assumptions of fixed duration of motion time was also introduced by Pazzi and Visioli [39]. The minimum-jerk trajectory optimization algorithm that was presented differs from the former study [38] due to minimizing the maximum absolute value of the jerk profile along a given trajectory. Trajectory was formulated by means of a set of cubic spline functions. An optimization algorithm was defined by the interval analysis method, which guarantees that the global minimum point [40] is achieved in their proposed method.

2.1.3 Energy Optimality Optimization

Another trajectory optimization criterion, where the cost function of the optimization is the minimization of the actuator torque and/or energy required by the robotic manipulator is also taken into account in the literature. The topic of optimal energy criterion for the desired task has also been a crucial research area since 1970's [41]. A lot of investigations and demonstrations have been carried out to optimize the desired manipulator's trajectories based on the concept of minimum-energy consumption.

However, minimum cycle time is a popular optimization criterion for most industrial robotic implementation to minimize the production cycle and to increase the profit of the desired production [6].

In the trajectory optimization literature, although the minimum cycle time is a popular optimization criterion and dealt with that more than the minimum energy criterion the minimum-time criterion is not convenient if a smooth trajectory for the desired motion is required. High cycles in the actuators can result in physical oscillation and undesirable shocks to the manipulator's structure which is as a result of these, a wide range of undesirable problems will occur including loss of accuracy, increased energy consumption and a decrease in actuator life span [35]. Moreover, reducing energy consumption may be a desirable criterion in numerous implementations, such as; robots for space (Mars robot "Curiosity"), submarine exploration, or unmanned reconnaissance vehicles, repetitive applications [35], where the success of the desired mission depends extremely on conserving available energy for the desired task.

In the repetitive robotic implementations in the industry, a huge amount of electrical energy/power is needed for the large-scale robotic manipulators during the desired implementations. For instance, approximately 15,000 robotic manipulators were installed in 2004 in North America [5]. If 10 kW-hr is consumed by each robotic manipulator, which works continuously for 24 hours shifts each day, the consumed electrical energy/power will be significant for 24hr. If the energy consumption can be reduced by 10 percent, then it would be the huge impact on the environment and cost. When repetitive implementations are executed, it is possible to develop a method in order to

move a robotic manipulator along a specified optimum trajectory with minimum cost by taking into account the mechanical energy and/or power for the formulation of the cost function.

In the literature regarding minimum-energy criterion, the energy minimization algorithm was considered by a dynamic programming search method which considered the dynamics of the robotic manipulator for point-to-point motion profiles by Vukobratovic and Kircanski [42] and for movements along the given manipulator trajectory by Shin and Mckay [43] subject to input torque/force constraints.

Another point-to-point trajectory motion profile for an industrial robotic manipulator of the type Manutec r3 was taken into account by optimizing an energy criteria by Stryk and Schlemmer [44]. In their proposed study, the optimal solution was observed by converting the optimal control issue into a boundary value problem, which was later computed by the trajectory optimization algorithm for the three degrees-of-freedom robotic manipulator. A trajectory optimization algorithm was formulated by the combination of direct collocation and indirect multiple shooting [45] to handle the variables and the system constraints effectively and to obtain the optimal trajectory solution accurately. A multi-objective cost function and the system constraints were handled by the direct collocation procedure in the trajectory optimization algorithm. In their proposed work, a minimum-time trajectory criterion was also considered for the desired trajectory motion. However, the violation occurred on the velocity profile of the robotic manipulator's during the large parts of the minimum-time motion. The numerical outcomes verified that the vibration on the robotic manipulator's actuators was significantly decreased by utilising a minimum-energy consumption trajectory, which was only approximately ten percent slower than the minimum-time trajectory.

The reason of preferring the computationally efficient optimization methods in any trajectory optimization algorithm is that a large number of variables and constraints may adversely affect the outcomes of optimization and computational efficiency. In Field and Stepanenko's research [46], in order to handle the extremely complex functions and the system constraints easily, the minimum-energy consumption trajectories were formulated by iterative dynamic programming by taking into account joint actu-

ator profile and time constraints of the desired motion. Although a global optimum solution was not achieved in their optimization program, the program has the ability to prevent some poor local minima points during the trajectory optimization process. The iterative dynamic programming method comprised of a parallel structure that results in significant computation time reduction on parallel connected computers. Uniform cubic B-spline functions [47] were utilised to identify the point-to-point trajectory profile for the trajectory optimization algorithm.

Another energy-optimal trajectory was determined by a constrained parameter optimization over a set of cubic B-spline functions by Martin and Bobrow [48]. In their proposed approach, the energy-optimal problem was converted into a discrete parameter trajectory optimization problem by parameterizing the joint trajectories. The cost function was formulated by means of the time integral of the squared joint torque profiles.

For the minimum-energy consumption algorithms, the instantaneous power in the actuators is the product of instantaneous joint torques and joint velocity profiles was taken into consideration by some researchers [49], [50]. Their proposed trajectory optimization algorithms ignore the fact that a large amount of energy is dissipated on the joint actuators of the mechanical system even though the robotic manipulator's links are constant. In order to eliminate these limitations, the integral of the sum of the squares of the joint torques as a dissipated energy/power criterion was taken into account for the optimum energy consumption algorithm [51].

Another dissipated energy/power criterion was taken into account in order to formulate a cost function by Garg and Kumar [52]. Genetic and simulated annealing optimization algorithms were compared on multiple robotic manipulators in order to identify an optimal trajectory. The optimal trajectory was determined by utilising both techniques, which were based on minimum-energy consumption requirements. A fourth-degree polynomial function was utilised to define the joint trajectory. It was found that both trajectory optimization techniques were converged to the optimum solution, however, simulated annealing reaches the optimum solution faster than genetic algorithm method (GA).

Another GA trajectory optimization approach was introduced by Biswas, Deekshatulu and Roy [53]. In their proposed research, the collision avoidance method with optimal trajectory planning of a three degrees-of-freedom robotic manipulator structure was investigated. Actuator torques and collision avoidance were considered for the formulation of the cost function, and a forth degree polynomial function defined the trajectory of the robotic manipulator. A multi-objective genetic optimization algorithm method was used, which minimizes the torque in the actuators and maximizes the distance of end-effector to the obstacle simultaneously.

In this case, utilisation of an optimization solver such as a built in function may be desirable in order to reduce the computational time for creating essential trajectory optimization algorithm routines. Hence, another trajectory optimization method is introduced in the scientific literature, such as the sequential quadratic programming method (SQP) for the unspecified execution time and the unknown manipulator position of the control points by Chettibi and Lemoine [54]. A set of spline functions were utilised to define the desired trajectory. In their proposed trajectory optimization algorithm, the formulation of the cost function consisted of electric power consumed for each manipulator's actuator for the non-linear optimization program subject to the electro-mechanical system constraints.

The limits of the position, velocity, acceleration, obstacle avoidance, singularity avoidance and torque values, etc., can be shown as the constraints of the robotic manipulators. The unrealistic or unreachable movement of the robotic manipulator can be avoided by utilising these constraints in the trajectory optimization procedure. In some cases, (Rubio [30] and Kagan, Iravani and Sahinkaya [9]) the cost function calculation can involve running the time consuming inverse dynamic model. In conventional methods, (such as SQP), the constraints equations are handled separately, and the cost function is called regardless of whether the constraints are satisfied or not. Therefore, in order to improve computational efficiency Kagan, Iravani and Sahinkaya [9] proposed an optimization method which handles the constraints within the cost function calculations and the inverse dynamic analysis is only evaluated when these constraints are satisfied. Trajectory of the robotic manipulator is represented as a fifth order

B-spline function. The parameters of the B-spline function were optimized using a multi-parametric optimization algorithm and actuator torques have been considered for the formulation of the cost function, which utilises an inverse dynamic analysis. By taking into account the kinematic and dynamic constraints to be included in the cost function [9], the complexity and computational effort of the optimization algorithm were significantly reduced.

2.1.4 Multi-Objective Optimization

The implementations of today's industrial sector consist of multi-task industrial issues. The various complex design specifications and dynamic and kinematic constraints have to be handled effectively during the desired implementations which involve simultaneous consideration of multi-cost criteria that search for the optimal outcome within a feasible decision variable space [55] in this kind of complex environment. Therefore, the designers will decide on what is achievable for implementation by taking into account these optimal solutions from a multi-cost optimization method, which let them make an exactly informed selection for the intended task and productivity is increased by taking into account this optimal solution in the industrial processes. In this section, types of multi-cost methods for the trajectory optimization procedures have been given.

2.1.4.1 Optimum Time and Jerk

In some literature, the time-jerk optimum manipulator trajectories can be defined as another way of tackling the optimum trajectory planning problem. That is, a cost function was formulated by two terms to be minimized; a first term can be given as the proportional to the executing time and the derivative of the acceleration can be defined as the second term [56]. This hybrid cost function created opposite effects to make them work against to each other for the intended system [56]. For instance, reducing the execution time would inherently increase the values of velocity, acceleration and jerk profile, while minimizing the jerk guarantees the smoothness during the intended motion. It has to be noticed that the effect of the jerk profile is crucial if the accuracy is essential in the desired task. In this hybrid cost function, stability between speed and

smoothness can be achieved by suitably modifying the two terms of the cost function for the intended motion.

Gasparetto and Zanotto [57] also introduced a hybrid cost function which was formulated by two terms to be minimized, time and jerk profiles, respectively. The sequential quadratic programming (SQP) method was utilised for the optimization algorithm by taking into account the constraints, which were defined as maximum and minimum limits on the absolute values of velocity, acceleration and jerk profiles for all robotic manipulator joints and they were set before execution. A six-dof robotic manipulator (a Cartesian robotic manipulator with a spherical wrist) was utilised to run the proposed trajectory optimization method and a fifth-order B-spline function defined the robotic manipulator's trajectory. This work followed from the previous works reported in [58], [59].

The above works extended by Gasparetto, Lanzutti in [60], where an experimental evaluation and validation was carried out an two optimum time-jerk trajectory planning algorithms and compared with a global optimum jerk profile with cubic spline and fifth-order B-spline algorithms. In their trajectory optimization algorithm, the requirement for a fast implementation or the requirement for a smooth manipulator trajectory movement can be achieved by modifying the values of two weights in the hybrid cost function.

2.1.4.2 Optimum Time and Energy

In addition to the hybrid time-jerk cost functions, a cost function can also be constructed by taking into account both optimal execution time and the minimum-energy consumption of the actuators for the trajectory optimization algorithm. For instance, a method for computing time-energy optimum manipulator trajectories along the specified path has been introduced by Shiller [61]. In his research, the full non-linear system dynamics and actuator constraints of the mechanism were taken into account in the optimization problem and the initial value of the Lagrangian multiplier was iterated in order to solve the proposed optimization method. The time-energy hybrid cost function provided better transient response and smaller tracking errors than the minimum-time

optimal trajectories [61]. A two link planar robotic manipulator and also the UCLS direct drive robotic arm were utilised for theoretical and experimental studies in order to verify the effectiveness of the proposed trajectory optimization method.

Another instance of time-energy optimum trajectory was demonstrated by Saramago and Steffen [62]. The trajectory optimization problem consisted of moving a robotic manipulator with minimum time-energy hybrid cost criterion along a specified geometric trajectory subject to kinematic system constraints on the maximum values of velocity, acceleration and jerk profiles. The optimum result was observed by modifying the two terms in the cost function for the optimization algorithm, which considers a manipulator dynamics and collision avoidance scheme. Three and six degrees-of-freedom of robotic manipulators were taken into account to verify the effectiveness of the proposed method.

These methods were extended to the presence of movable obstacles [63], where a multi-criterion cost function was also utilised in order to get optimum trajectory. Sequential unconstrained minimization techniques were utilised for the formulation of the optimization algorithm. A set of cubic B-spline functions were used to define a point-to-point motion trajectory. The non-linear manipulator dynamics, actuator constraints, joint limits and obstacle avoidance were taken into account by the trajectory optimization algorithm. In order to guarantee collision free motion between the robotic manipulator and obstacles, the penalty function was inserted to the conventional cost function profile. The system constraints were utilised to characterize the minimal collision free distance between the robotic manipulator part and surface of the obstacles, therefore, the penetration by the end-effector of the robotic manipulator into a restricted area around the obstacles was avoided.

Another minimum time-energy criterion with collision avoidance scheme for robotic manipulators was introduced by Chettibi and Lehtihet [33]. A minimum time-energy trajectory problem was calculated by taking into account the kinematics and dynamics of the robotic manipulator performance for the intended task. In their study, the trajectory optimization problem was extremely non-linear due to the complexity of the robotic manipulator dynamics and the non-linearity of the multi-cost function and the

system constraints. In their study, a non-linear trajectory optimization was generated from a trajectory optimization problem by formulating the joint temporal position via the uniform cubic spline function. In order to calculate the hybrid cost function (such as transfer time, and consumed energy of the joint actuators), a Sequential Quadratic Programming (SQP) algorithm was utilised. Smooth motion was executed by selecting weighting factors as demonstrated in [62].

However, any non-linear constraint trajectory optimization method such as Sequential Quadratic Optimization algorithm, can result in a local minimum point. One way to overcome this problem is either the trajectory optimization algorithm can be initiated from various initial guesses or a genetic optimization algorithm and/or any other global solution technique can be utilised in order to scan the feasible solution space and to achieve to a global minimum point. Ramabalan and Saravanan presented a multi-objective dynamic optimal trajectory planning based on a genetic algorithm technique [64]. The resulting optimized trajectory was ensured as smooth and fast enough by their proposed optimization technique, in addition to this, global optimum solution was achievable. Both kinematics and dynamics system constraints and also collision avoidance constraints were taken into account during the trajectory optimization process.

Another global optimum point for the intended trajectory was achieved by Xu and Zhuang [65]. The corresponding proposed trajectory optimization model was based on Environment-Gene Evolutionary Immune Clonal Algorithm, which has the capability to achieve the global optimum point for the intended task. The effective global solution was provided due to a novel ranking technique for the penalty function. The achievable motion was continuously checked by the penalty function in order to prevent ending in a local minimum point during the trajectory optimization process. A set of cubic splines were utilised to define the manipulator trajectory. Also, kinematics and dynamics system's constraints were taken into account by means of upper bounds on velocity, acceleration, jerk and input torques profiles of the actuators.

2.1.4.3 Other Multi-Cost Criteria

In order to achieve the intended motion in the industry, the capabilities of the industrial robotic manipulator have been restricted due to the limits of maximum acceleration and maximum deceleration profiles along the solution curve, many undesirable points such as singular points, local minimum point, critical point for the robot configuration and obstacles. All these restrictions in practice lead to failure in finding the optimal solution for the trajectory, or spend a lot of time and memory to determine optimum solution.

To overcome these problems, a non-linear constrained trajectory optimization procedure based on an evolutionary algorithm (elitist non-dominated sorting genetic algorithm (NSGA-II)) and a differential evolution (DE) algorithm for fixed, mobile and oscillating obstacles were introduced by Saravanan and Ramabalan [66]. The combined multi-criteria cost functions were formulated by means of 6 optimization criteria which are duration of motion, actuator torque profile, penalty for collision free motion, singularity avoidance, joint jerk minimization and optimal joint acceleration profiles subject to the kinematics and dynamics of the system constraint equations. The robotic manipulator's trajectory was formulated by B-spline functions. With this trajectory optimization algorithm, a global optimal solution is possible and the method is applicable for all other types of robotic manipulators.

Same trajectory optimization algorithm without collision avoidance criterion [67] can also be utilised for free motion planning for the robotic manipulator or a measure of manipulability [68] can be inserted to the trajectory optimization algorithm. A measure of manipulability has useful features for the robotic manipulator design, task planning, and faster recovery capability from the escapable singular points for the robotic manipulator.

2.2 Trajectory Optimization for Redundant Manipulators

Most of the commercial non-redundant industrial robotic manipulators consist of six degrees of freedom in order to manipulate the object's position and orientation dur-

ing the industrial implementations. Although the non-redundant commercial robot's performance and capability provide significant advantages for industrial implementations, however, some other complex tasks need to be handled with better efficiency and flexibility (such as drilling, cutting, medical robotics, maintenance of nuclear reactors etc.). In order to meet these industrial demands, robotic manipulators should consist of more degrees-of-freedom than necessary to perform complex intended tasks effectively and successfully like human arms [69]. Therefore, the research on redundant robotic manipulators has increased rapidly and become a highly attractive area in order to investigate the capabilities and advantages of redundant robotic manipulators.

Redundant robotic manipulators have many significant advantages over non redundant ones and other performance criteria can also be handled effectively by utilising redundancy such as collision avoidance [69], [70], [71], [72] in working space, preventing singularities [73] where the manipulator lose some degrees-of-freedom, avoiding limits of joints [74], optimizing some cost function such as reducing torque/energy [75], minimal joint motion [76] over a intended task and also ensures high degree of manipulability during the execution of the desired motion of the end effectors [73].

Redundancy can also be utilised for fault tolerance during the intended task [77], [78], [79], [80], [81]. For instance, when a failure has occurred in the non-redundant robotic joint that automatically results in the loss of full end-effectors controllability. On the other hand, the extra degrees of freedom can be introduced in order to compensate for the manipulator's failure in the kinematically redundant robotic manipulators [82]. By this way, even if the end-effector's position remains unchanged, the rest of the robotic arm will be free to move by utilising these extra degrees of freedom.

These kinds of crucial issues such as singularity avoidance, collision avoidance and torque/energy optimization in non-redundant manipulators become less difficult when introducing the redundant robotic manipulators [82]. In this section, a literature review of the redundant manipulators is given with respect to the optimization criteria.

2.2.1 Singularity Avoidance

One of the motivations for introducing redundancy can be demonstrated to handle the singularity avoidance for the robotic manipulator. Singularity is an inherent problem which may be occurred at any time within the robotic manipulator's workspace and robot's control [83], and it may be indirectly considered as part of the optimum trajectory planning problem.

In the literature, many researchers have broadly investigated the singularity problem for the case of non-redundant robotic manipulators [84], [85], [86], [87] and also redundant robotic manipulators [75], [69], [73], [88].

Usually, when a robotic manipulator's end-effector is close or/and near to its reachable point of its workspace, a singularity may occur any time in its kinematic mapping. At a position close to the point of singularity, matrix inversion of the robotic manipulator becomes ill-conditioned [89] and, as a result of this, either the control algorithm collapses or the joint velocities and acceleration profiles required for the proposed robotic trajectory may have unsustainably large values or unlimited for relatively little end-effectors movement [90]. In this case, at least one DOF will be lost in the robotic manipulator and the motion of the robotic manipulator in this direction will become unfeasible [91].

Singularities may result in:

- Loss of freedom
- Reduction of Workspace
- Loss of control

In the literature, to prevent singularities for the robotic manipulator, a manipulability measure for redundant and non-redundant manipulators was demonstrated by Yoshikawa [86] and a trajectory control algorithm was developed by taking into account this measure technique and the redundancy.

Another method of handling singularity is the damped least-squares solutions introduced by Nakamura and Hanafusa [92] and Wampler [93]. In their method, a desired

task-space trajectory was converted to a desired join-space configuration of the manipulator and then a joint-space controller was utilised in order to realize the desired joint-space configuration [94].

Another singularity avoidance study was performed by Lee and Lee [95]. For handling the singularities robustly, a torque optimizing control method was proposed in their study. The weighted generalised inverse and the damped least squares inverse of the Jacobian-inertia product were utilised to formulate and solve the dynamic control equation for the end-effector of the robotic motion control. By introducing a damping factor as a function of the generalized dynamic manipulability measure [96], [97], the end-effector acceleration deviation due to the damping factor was minimized. The generalized dynamic manipulability measure was executed in order to characterize the robotic manipulator's performance in terms of achievable end-effector acceleration profile subject to the weighted constraint on the joint torque profiles.

2.2.2 Obstacle Avoidance

Tracking the intended end-effector's trajectory accurately and effectively is a primary objective of the industrial robotic controller design. Redundancy in the robotic manipulators does not only satisfy the tracking issue, but also collision avoidance in working space can also be shown as a secondary purpose for the intended task [98]. The research of collision avoidance can be shown as an active area in the robotic literature and a vital part of the successful trajectory planning for the intended task [99].

Although various collision avoidance methods are introduced in the literature, off-line trajectory planning method (high level of the manipulator control hierarchy) and on-line trajectory planning method (low level of the control hierarchy) [72], [99], [100], [101] can be shown as the two main collision avoidance methods for the robotic manipulators.

Most of the off-line trajectory planning method is computationally expensive, time consuming and restricted to structured environments and fixed obstacles [99]. A curve profile can be generated by mapping the trajectory of the robotic motion and the collision free trajectory into a curve profile in the Configuration space (C-space). The

C-space consists of a rectangular co-ordinate space where each axis represents one joint parameter and many trajectory planning algorithms for numerous robotic manipulators utilised C-space. So, descriptive data of the free zone of the workspace will be searched by off-line trajectory planning method of collision avoidance. This implementation is time consuming since this method is based on the convex sets in the Configuration space (C-space). If the system consists of a high-DOF, off-line method is not convenient and applicable for real time collision avoidance scheme. This is because it will be very hard to visualise the robotic manipulator in C-space due to the robotic manipulator configuration becoming a point without link lengths [82], [102].

On the other hand, artificial potential fields were utilised [103] for the concept of on-line method, which is an energy based approach. This concept is computationally more efficient and convenient for the implementation of unstructured dynamic environments as well as mobile obstacles. Artificial potential fields keep the robotic manipulator away from each obstacle's surface, and it was first introduced by Khatib [100]. A substantial feature of his research is the demonstration of obstacles, which were generated by basic geometric shapes for the intended motion.

In online trajectory planning method, high potential energy indicates the obstacles while low potential energy is used to show the free areas of the workspace [101]. Therefore, the robotic manipulator configuration takes into account the potential fields in the working space in order to prevent the collision.

Another artificial potential field technique has been established by Khatib [104]. In his research, the obstacles were indicated by repulsive surfaces and an attractive pole was utilised to indicate the position to be reached for a given task. In this case, robotic manipulators can reach the intended position without collision. However, the implementation of this proposed potential field approach was limited due to the existence of local minima point for the desired robotic manipulator's trajectory, and it was not capable of handling the random shape obstacles.

In order to overcome these restrictions, the harmonic potential function was introduced by Kim and Khosla [105]. However, the computational complexity of their proposed technique was intensive, especially when the robotic manipulator has larger

DOFs.

Another collision avoidance study was introduced by Maciejewski and Klein [98]. In their approach, each time the point on the robotic manipulator was identified that was the shortest distance to an obstacle surface, termed as the collision avoidance point, and a desired velocity component in a direction that is directly away from the obstacles surface was assigned to it. In this case, the primary purpose of the algorithm was specified by end-effectors velocity profile and the secondary purpose was collision avoidance velocity profile. Their proposed approach worked well in a workspace with relatively few obstacles and having a large free area.

In Baillieul's research [106], the collision avoidance problem was also taken into account for three-bar type redundant robotic manipulators. Basic geometric obstacle shapes were also utilised in order to display obstacles as in the work of [100]. In order to prevent collision with obstacles, extended Jacobian method was utilised as an alternative collision avoidance technique. The kinematic redundancy of the system was resolved by imposing additional constraints to constrain the robotic system completely. As it was mentioned before, in order to simultaneously handle the primary and secondary purposes, algorithmic singularity problems were likely to happen [98]. Therefore, algorithmic singularity may be also caused by this alternative collision avoidance method to induce infeasible solutions even though the end-effectors Jacobian has no singularity [106].

In order to handle this algorithmic singularity issue, the constraints were included for collision avoidance restricted joint range. Hence, the Jacobian metric extended and improved by Sciavicco and Siciliano [107]. It has been clearly shown that, the collision with an obstacle was prevented by utilising these constraints and/or the violation of a mechanical boundary on a joint variable was avoided. Similarly in [106], the algorithmic singularity may still occur.

The quadratic programming approach (QP) was introduced based on the pseudoinverse method to resolve the redundancy of the robotic manipulator subject to the inequality constraints like joint limits and joint velocity limits for the inverse kinematic control of redundant robotic manipulators [108], [109], [110].

Another collision avoidance method was introduced by Guo [109]. In his study, the distance between the robotic manipulator links and the obstacle's surfaces was maximized in order to prevent the collision with obstacles. Although the robotic manipulator was far away from the obstacle's surface, the robotic manipulator's redundancy for distance maximization was unnecessarily determined in the program. This redundant computational process is time consuming and a heavy computational burden for the collision avoidance algorithm. Therefore, it was not convenient for the real time implementations. In another collision avoidance approach [108], the dynamic equality constraint of the mechanism for the collision free criterion was taken into account.

In Sezgin and Seneviratne's work [111], two collision avoidance implementations for serial redundant robotic manipulators were introduced for predetermined end-effectors trajectory. In the first approach, a single obstacle is introduced in the workspace, and the sum of the instantaneous space between the configuration control points on the robotic manipulator and the obstacle's surface was maximized. In the second approach, multiple obstacles were introduced in the workspace, and the robotic manipulator went between them. However, their approach was applicable for only two-dimensional robotic manipulator systems and also dynamic environments where the obstacles are no longer fixed.

For most of the research mentioned above, the redundancy in the presence of obstacles was resolved at the joint velocity level [69], [98], [106], [108], [111]. Unlike the past researches, in Ding and Ong [71] approach, the redundancy was resolved at the joint position level. Therefore, this approach can be easily utilised in the commercial industrial robot controller. The basic concept of the resolution scheme was to keep as far away from the obstacles surface as possible. Therefore, the safest configurations are always provided due to a resultant pose of the robotic manipulator. And as the collision algorithm was generic, and it can be easily implemented for preventing collision between link-obstacle and link-pairs. Unlike Sezgin and Seneviratne's work [111], the proposed method was applicable in both two and three dimensional spaces.

2.2.3 Other Optimality Criteria

Redundancy in the robotic joints has many important features such as preventing singularities, joint limits and obstacles and optimizing a variety of optimization criteria while executing an intended end-effector motion. Other instances of performance criterion can be taken into account such as minimizing joint torque, which is a very appealing subject in robotics research.

Most of the research in the literature in terms of redundancy resolution has considered the kinematic and/or geometric level, primarily through instantaneous implementations of the pseudoinverse of the Jacobian matrix computations. However, the dynamic properties of the robotic manipulator are not taken into account by the utilisation of generalized inverses. In some cases, introducing kinetic criterion may also be desirable such as reducing the torque loading at the manipulator joints by appropriately resolving the redundancy. Therefore, a generalized inverse has to be generated in terms of the acceleration profiles to combine them into dynamics.

For instance, in Khatib's research [112], the inertia-weighted generalized inverse scheme was utilised and this research was followed by Vukobratovic and Kircanski [113] to incorporate an energetic model of hydraulic and electromagnetic motors, and implemented the resultant generalized inverse to velocity profiles.

In addition to the above research, the minimization of the torque loading at the manipulator joints was widely investigated by taking into account a variety of local optimization methods by [75]. In their proposed research, the null space algorithm of the Jacobian was utilised to derive a torque optimization solution. The joint torques were optimized instantaneously by this algorithm such that the joint torques were placed closest to the mid-points between the maximum and minimum joint constraints. Interestingly, their research indicates that all of these methods led to instability issues in the robotic manipulator motions for many trajectories even though locally they reduced the torque magnitude in the joint actuators. The longer manipulator's trajectories also had these unstable motions because the torque limitations were greatly exceeded by whipping actions that thrust the endpoint of the intended trajectories.

The local versus global trajectory optimization methods of minimizing torque magnitude for the joint actuators were examined by Suh and Hollerbach [114] in order to prevent the instability problem during the intended motion. In their study, simulations demonstrated that unexpected instability problems for relatively long manipulator trajectories were occurred in the local trajectory optimization methods. However, although a global trajectory optimization method was not suitable due to computational complexity for real time control and implementations, outcome of the global solution was highly promising, and a stable solution was determined and was better than the results of the local algorithms in movements of all lengths, even those long trajectory movements where the local trajectory optimization methods demonstrated the instability problem.

A local joint torque minimizing pseudo-inverse approach was derived by Nedungadi and Kazerounian [115] instead of taking into account the null space of the Jacobian approach. However, both approaches were found to exhibit instability problems for a long-range intended trajectory motion.

In addition to all these works, in order to determine the cause for instability problems and/or to formulate other local joint torque minimization algorithms, many studies have been introduced by researchers [116], [117], [118]. However, all these investigations collapse due to instability problems in the joint torque profiles.

Another joint torque minimization method was executed by Kang and Freeman [119], who introduced the null space damped joint torque minimization method in which the local joint torques were stabilized by damping forces created from convenient null space.

The instability issue of redundant joints of the robotic manipulator was also addressed by Ma and Nechev [120]. In their proposed research, they demonstrated that local torque optimization techniques collapse in some of the research [116], [117], [118]. The reason for this is that the control over the homogeneous joint velocity profile was not considered by them due to the incorrect initial formulation of the optimization problem. By examining the self motion dynamics, two null space acceleration components of opposite direction were demonstrated. The recent stable scheme about satisfactory

torque performance of techniques was clarified by this fact, based on the joint velocity minimization.

In order to stabilize the joint torque profile, a balancing technique was also presented by Ma [121]. In his study, the solution of local joint torque minimization was balanced against the solution of joint velocity minimization by considering a weighting factor. In this case, building up high joint velocity profiles for a manipulator was avoided by the approach of joint velocity minimization, and therefore, resulted in stable arm motion.

For an optimization issue, the trajectory planning problem for redundant manipulators was also taken into account by reducing a dynamic cost function. For example, a cost function was formulated by the input electrical energy/power and trajectory deviation for a redundant manipulator, which was investigated by Hirakawa and Kawamura [122], [123], [124]. The variational optimization approach by using B-spline trajectory was introduced for optimizing the consumed electrical energy/power of a robotic manipulator system. In order to produce a powerful approach to optimum motion planning for redundant robotic manipulators, the use of B-spline trajectory generation with the use of a steepest gradient trajectory optimization method were combined. However, in order to utilise the proposed approach, the determination of gradients of the cost function is required.

In another study, instead of utilising the consumed electrical energy/power as an optimization criterion, optimal torque criterion was taken into account for kinematically redundant robotic manipulators between two joint configurations by Wu [125]. The utilization of an iterative procedure in order to transform the minimum torque trajectory planning problem into a nonlinear programming problem was the difference between the traditional pseudoinverse control approach and his study. This approach was easily generated by a series of feasible solutions for the nonlinear programming [125]. In order to prevent the local minima point in the trajectory optimization algorithm, different initial starting points for a three link redundant robotic manipulator were utilised for the nonlinear trajectory optimization technique. It should also be pointed out that the proposed method can also be implemented to non-redundant robotic manipulators [125].

A quite effective non-linear trajectory optimization search method can be provided by genetic optimization algorithms [31] for complex high DOF multi-link solution spaces. A genetic optimization method can be defined as a stochastic search algorithm which can optimize nonlinear functions by taking into account the mechanics of natural genetics and natural selection genetic optimization algorithms [31]. Unlike the conventional optimization algorithms, genetic algorithm does not take into account the computation of gradients, Hessian matrices and so on.

Indeed McAvoy and Sangolola [126] have demonstrated that a genetic optimization algorithm provides optimal solutions for the constrained nonlinear trajectory optimization problem. In their study, B-spline trajectory generation and genetic optimization algorithm were taken into account by the proposed trajectory optimization algorithm for the solution of the trajectory optimization problem of optimal pick and place task planning for redundant robotic manipulators. The cost function was formulated by a weighted sum of an integral of the end-effector's trajectory deviation term and a total joint actuator energy term (integral of power). The choice of weighting factor in the cost function was also argued by them. For example, if the value of weighting factor is increased, the robotic end-effectors will track the trajectory more accurately [126]. However, this result increased the total energy required for the desired motion.

In all of these approaches mentioned above the joint torque limits in the trajectory optimization algorithm were not taken into account. The actuator limits may be exceeded by the computed driving torques if the robotic manipulator is to handle the heavy items. In this case, the torque saturation may occur by exceeding demand torques. This results in a reduction in tracking precision for the intended task. In recent years, another trajectory optimization method, which is a neural network method, has also been addressed for the redundancy resolution of the robotic manipulators [127], [128], [129], [130], [131].

A method was introduced by Tang and Wang [132] in order to prevent torque saturation for real time joint torque optimization. Two recurrent neural network approaches were utilised in the proposed method in order to determine the driving joint profiles of the robotic manipulators directly. Therefore, the torque saturation was avoided in

real time implementation. In this proposed method, the first neural network was called the Lagrangian Network method and the second was called the Primal Dual Network method. The computation schemes were utilised for both of the optimization algorithms. In order to drive the robotic manipulator, the desired acceleration profile of the end-effector was utilised as an input to the optimization algorithm to get the signal of the optimum driving joint torque profile. Both of the algorithms were capable of generating a minimum stable driving joint torque profile.

The joint torques or force profiles are utilised to control the robotic manipulators in order to provide the required acceleration profile. Optimizing joint torques can be more attractive than optimizing joint velocities or acceleration profiles [115], [116]. For this reason, a control algorithm with local torque optimality and singularity robustness for redundant robotic manipulators was introduced by Chung and Lee [133].

Khoukhi and Demirli [134] presented a multi-level hybrid neuro-fuzzy optimization method to multi-criterion cost functions for the trajectory planning of an n-DOF planar serial robotic redundant manipulator by taking into account workspace limitations and task requirements. Optimum execution time, optimum energy consumption and singularity avoidance were considered for the formulation of a multi-criterion cost function. In their study, the trajectory generation procedure consisted of two stages. An offline trajectory planning generating a large dataset of multi-criterion cost trajectories with respect to their workspace limitations was taken into account as a first pre-processing stage. In the second stage, an augmented Lagrangian technique optimized these trajectories to design optimal trajectories for online programming. The second contribution consists of a systematic way for create a neuro-fuzzy model to online multi-criteria cost trajectory planning upon outcomes of simulation data generated from a complete dynamic model of the robotic manipulator and related constraints.

Another non-conventional trajectory optimization method was studied by Arakawa [135]. Optimum energy consumption of the redundant joint actuators was taken into account for the formulation of the cost function, which is used by a virus-evolutionary genetic optimization algorithm. A host population and virus population methods were composed with subpopulations in order to prevent the collision in the intended

task [135].

Marcos and Machado [136] also utilised multi-criterion cost functions which consisted of joint displacement minimization, velocity minimization, acceleration minimization, minimizing joint torque and total joint power. The closed loop pseudoinverse methods with genetic optimization algorithms were combined for a new technique that was executed in order to optimize the intended trajectory of the redundant robotic manipulator.

Evolutionary search optimization algorithm was utilised for another global optimization approach by Sullivan and Pipe [137] in order to reduce the jerk profile in the desired trajectory. Global trajectory optimization approaches based on minimizing a jerk profile were demonstrated effectively and advantages of the proposed method were shown in terms of preventing singularity and stability, when compared with a local trajectory optimization algorithm based on instantaneous criteria [138], [139].

Ata [140] also reduced the jerk profile of the robotic manipulator in order to achieve a soft motion trajectory for an intended task. The jerk profile was reduced by this soft motion trajectory, and the controllability of the intended job was increased; in particular if the end-effector holds a liquid stuff. In his study, a comparison between the soft trajectory and the normal linear segment with parabolic blends trajectory (LSPB) was demonstrated [140]. The result shows that the soft motion trajectory has advantages over the LSPB such as the jerk profile which can be fully handled in the soft motion trajectory although there is a vibration problem at the start and end of the LSPB trajectory.

Human motion planning based on the optimal jerk model is a similar issue to robotic manipulator trajectory planning, which needs to be considered with optimal jerk profile in order to decrease the vibrations and improve control performance during the given task. Another smooth trajectory for the end effector was determined by the minimum jerk model in Cartesian space by Yang and Kim [141]. Each of the robotic joints was formulated by means of the third degree B-spline curve with unknown parameters (control points). A general method for predicting joint profiles was executed, and this was broadly applicable to both linear (straight) and non-linear (curved) trajectories

of robotic manipulators. A dynamic effort of the robotic manipulator, joint angle consistency, which is the joint range change (first derivative), and acceleration function were also taken into account in the cost function, and robot execution time was preferred as a user-assigned term rather than a part of the cost function [141]. The mathematical formulation of their proposed method can be utilised for robotic manipulators with any number of degrees of freedom.

Another trajectory optimization criterion for a time-optimal motion was also taken into account for kinematically redundant robotic manipulators [142]. A method was introduced to figure out the optimum minimum-time trajectory planning problem of serial redundant robotic manipulators which were executed by utilizing the phase-plane analysis and linear programming algorithm. In order to demonstrate the validity of the proposed scheme, a three link planar rotary redundant robotic manipulator was simulated.

Some optimization methods consider minimizing the deviation of end-effector positions in the trajectory, e.g. [143]. In order to track the prescribed manipulator's trajectory accurately, the sum of the position deviation of the end-effectors of each intermediate point along the manipulator trajectory was reduced for the main purpose of his study. A trajectory optimization algorithm was combined with a genetic algorithm and pattern search technique. Three degrees of freedom planar redundant manipulator with various end-effectors trajectories were simulated to verify the proposed trajectory optimization algorithm. In their optimization algorithm, optimal points from genetic optimization algorithm were introduced into pattern search algorithm as inputs to improve the results [143]. That is, the genetic optimization algorithm was utilised to search for the global minimum point for the intended trajectory and the pattern search algorithm was utilised to refine the local minimum achieved by the genetic optimization algorithm. In some cases [144], reducing the positional deviation was utilised as a constrained trajectory optimization criterion while simultaneously preventing any collision of the redundant robotic manipulator with either the obstacles or within its links.

Another paper [145] develops an efficient control algorithm in order to avoid computational complexity of the redundant and hyper-redundant manipulators. The cost

function for an optimum path was based on minimum torque and/or energy consumption. In their study, optimization technique utilized an inverse dynamic analysis and used a fifth-order B-spline function for the desired trajectory. Unlike the other conventional optimization methods, the system constraints were handled within the cost function in order to avoid running the inverse dynamics when the constraints are not satisfied. A novel virtual link concept was introduced to replace all the redundant links to eliminate physical impossible configurations before running the inverse dynamic model [9]. The process was also applicable to hyper-redundant manipulators with a large number of links.

2.3 Trajectory Optimization for Hyper-Redundant Manipulators

A hyper-redundant robotic manipulator is a redundant manipulator and consists of a large or infinite number of degrees of freedom. The benefits of hyper-redundant robotic manipulators can be demonstrated as the capability to prevent obstacles, increased manoeuvring ability, and increased reliability due to mechanical failure [77], [78], [79], [80], [81], [146]. Therefore, hyper-redundant robotic manipulators can be conveniently utilised for complicated implementations where a standard commercial industrial robot could never perform [147], for example, maintenance of nuclear reactors components, [148]. However, their complex control mechanism and also their weight make them more difficult to utilise. If each joint arm has a driven mechanism, it will make the hyper-redundant manipulators heavier than usual.

Usually, a non-redundant robotic manipulator has minimal numerical complexity for the inverse kinematic solution. However, when a robotic manipulator has redundancy, it provides the capability for the robotic manipulators such as moving each joint in infinite ways for the same specified end-effectors motion. Therefore, despite their important features, each added redundant degree-of-freedom results in an increase in the complexity of the inverse kinematic solutions and control complexity of hyper-redundant robotic manipulators and trajectory planning problems also become

increasingly difficult [10].

The complexity of kinematic solution, long computation time, and difficulties in trajectory optimization can be demonstrated as disadvantages for inverse kinematic solution of trajectory planning for hyper-redundant robotic manipulators. In order to handle these disadvantages, the optimal solution with less computational effort for motion planning of hyper-redundant robotic manipulator was introduced by Yahya [149]. In some situations [149], a smooth trajectory for end manipulators was determined as the basic concept and this trajectory was identified by points which are close enough to each other. The most significant benefit of this scheme can be shown that when the end-effector of the manipulators moves to the next point on the trajectory, the angles between each closest links will be the same. In this case, the control of the motion of the system will be easier and more stable during the motion.

Another trajectory planning for highly redundant manipulators paper has been published by Conkur [150]. In this proposed research, B-spline curves already specified a path of the trajectory and this trajectory was tracked by each link of the robotic manipulator. During the motion, each link stays almost tangent to these B-spline path curves [150]. Therefore, each of the robotic manipulators moved around the given curve. That means that, the next link position was imitated by the most recently moved link in the next curve position. Control algorithm of the system consisted of three steps; first, the minimum space between link and the curve was specified, hence, control points on each link and their distance to the link are computed; second, link moved by taking into account the given control point; and three, all link positions have been checked by the algorithm [150]. Trajectory optimization technique was not addressed by this proposed scheme, but an efficient trajectory tracking algorithm was pointed out for the large number of links.

Animal structures are imitated by most of the recent mechanical systems such as the design of the hyper redundant robotic manipulator which can be compared to an elephant's trunk, snakes, tentacles [151] or a worm [152]. These kinds of animals consist of highly articulated structures, and provide skilful features such as the ability to go into a narrow and highly constrained environment while avoiding obstacles. As mentioned

before in section 2.2.2, obstacle avoidance is a crucial issue for robotic manipulators due to complete the required task effectively.

An obstacle avoidance algorithm for hyper-redundant robotic manipulators was presented in [153]. In this implementation, the obstacle avoidance problem was taken into account based on kinematic algorithms. The collision avoidance concept was defined as "*Tunnels*" in a workspace and a planar hyper-redundant robotic manipulator was utilised to implement the proposed method. A method of differential geometry was then utilized to formulate the equations in order to ensure that the robotic manipulator executed the motion in the tunnels without collision.

Another obstacle avoidance scheme for hyper-redundant robotic manipulators was proposed in [154]. The scheme was based on analysis in the defined posture space. Position of the end-effector manipulator's parameters in 3D was utilised to identify the hyper-redundant robotic manipulator configurations and then to describe the workspace of hyper-redundant manipulator with respect to the obstacles.

The generalized potential model was proposed in [155] to prevent collision and solve the trajectory planning problem for hyper-redundant manipulator in a 3D workspace. By utilising workspace information, repulsive force and torque between robotic manipulator and obstacle surface were determined by the proposed algorithm. By taking into account these forces and torques, the minimum potential configuration can be determined for the robotic manipulator. And then a collision free path can be achieved by locally modifying this configuration [155].

During the collision avoidance motion, control of hyper-redundant robotic manipulators contain many problems such as too many degrees of freedom, selection of task space, dynamic and also kinematic constraints to execute the system in real time.

Most of the proposed algorithm technique for dealing with the kinematics of redundant, hyper-redundant robotic manipulators that were suggested control the end-effector indirectly [156], through the rates at which the individual joints were driven, utilizing the pseudoinverse of the Jacobian. However, these proposed techniques lead to a kind of chaotic motion behaviour with unpredictable manipulator configurations for the redundant/hyper-redundant robotic manipulators. Motivated by this problem,

Duarte and Tenreiro [157] optimize the manipulability through a least square polynomial approximation in order to identify some joints positions.

In addition to that, a predictable motion of hyper-redundant robotic manipulators utilising constrained optimization control was introduced in [158]. A modular robotic manipulator was utilized to execute the desired motion. Angle, torque and also velocity limits of the actuator profiles were taken into account. A position goal, an orientation goal, minimal actuation energy, and actuation distribution were taken into account as individual goal objectives. Total cost function was shown as a weighted and normalised sum of the individual goal functions. The Matlab optimization function "$fmincon$" [159] was used for the optimization.

In another optimization approach demonstrated in [160], the closed loop pseudoinverse method was combined with genetic algorithms. Without combining the genetic optimization algorithm, pseudoinverse control was not repeatable and resulted in sliding on joint angles in the joint space configuration. The cost function was also formulated to minimize the joint displacement between the final and the initial joint position.

2.4 Concluding Remarks and Research Direction

Subsequent to this literature, there are a number of factors which have become apparent. Firstly, it is evident that various trajectory optimization techniques and also optimization criteria have been developed in order to determine the optimal solution for the non-redundant, redundant and hyper-redundant robotic manipulators. Each proposed trajectory optimization technique has provided a significant contribution to the trajectory planning problem, although some of them are not successful and capable of handling the given task.

As it is also seen from the various literatures, optimal minimum-time optimization algorithms have a crucial role to play in industrial productivity. However, this minimum-time criterion is not convenient for a smooth motion. This criterion may result in unwanted vibration and cause inaccurate end-effector motion and increased energy consumption. Due to this, minimum-energy consumption criterion has become

a crucial part of the optimum trajectory planning algorithms in industrial implementations. Successful implementations based on optimal minimum-energy criterion in the literature are quite promising for reducing the vibration profile and also reducing the energy consumption for the robotic manipulators.

Finding the optimal solution for any trajectory planning algorithm is a quite challenging procedure due to the large number of optimization parameters and coefficients may adversely affect the results of optimization, and computational efficiency. Therefore, these optimization parameters and also kinematic and dynamic constraints of the robotic manipulators need to be handled effectively during the optimization procedure. In addition to this, selection of the trajectory optimization technique is vital in terms of the success of any optimization procedure.

Although the non-redundant and redundant robotic manipulators have their own individual strengths and weakness, there is no doubt that the redundant robotic manipulators have many useful features over non-redundant robotic manipulators such as the capability of moving the joints in infinite ways for the same specified end-effectors motion. However, if the redundant manipulator has an infinite number of redundant links, the solution of inverse kinematic and control of redundant/hyper-redundant manipulators will become quite complicated and optimal trajectory planning problems also become unfeasible. Therefore, most of the research on redundant/hyper-redundant robotic manipulators concentrate on effective path following algorithms to handle the redundant link configurations.

Due to the above mentioned reasons, an efficient control algorithm will be developed in order to prevent computational complexity of the redundant and hyper-redundant manipulators. The important features of this control algorithm are that it can be applied to various robots, such as redundant, hyper-redundant or parallel robots in order to optimize the desired trajectory.

CHAPTER 3

THE EXPERIMENTAL SETUP AND MODELLING

An analysis of the literature concerned with various trajectory optimization techniques and also optimization criteria of the non-redundant, redundant and hyper-redundant robotic manipulators have been discussed in the previous Ch. 2.

This chapter details theoretical work carried out to create a simulation model of a 6-axis Neuronics Katana 450 6M industrial robotic manipulator, which is a Linux based robotic manipulator with 6 degrees of freedom with an operating space of about 602.4 mm from its flange. Each joint consists of a DC motor with incremental encoder, harmonic drive gears and an independent axis controller. The work presented in this chapter is utilised in the subsequent theoretical and experimental studies.

3.1 Description of the Industrial System

As mentioned earlier, the Katana 450 industrial robotic manipulator being demonstrated consisted of 6 degrees of freedom propelled by 6 DC motors with incremental encoder controlled by independent axis controller hardware.

An image of the six degrees-of-freedom of the Katana 450 robotic manipulators with end-effector tool is shown in Fig. 3-1, where the different parts of the mechanism (such as actuators, link lengths, angle limitations) are identified. The origin of the coordinate system is located at the point where the rotary axis of actuator 2 lier. The y represents

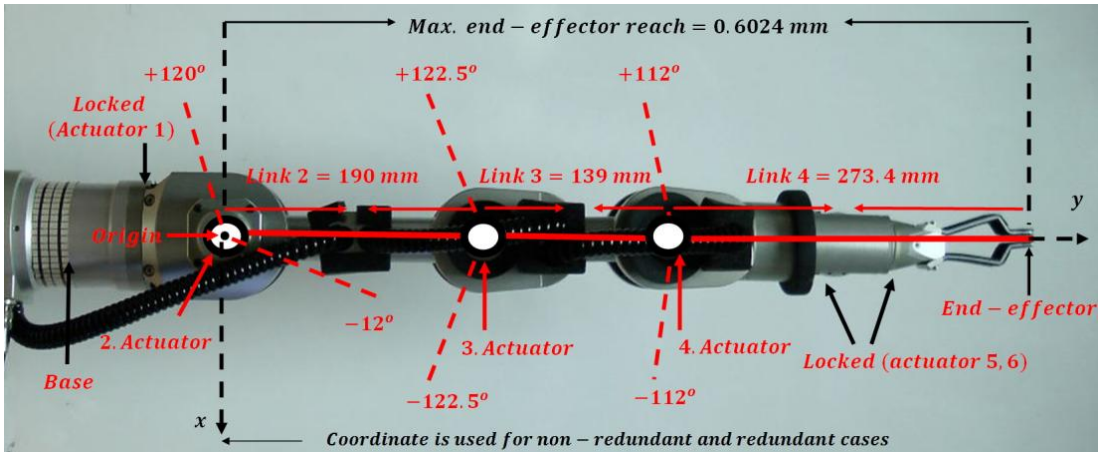

Figure 3-1: *Katana 450 industrial robotic manipulator.*

Parameters	**Link 2**	**Link 3**	**Link 4**	**Load**
Mass of Link (kg)	1.022	0.882	0.969	–
Length of Link (mm)	190	139	273.4	–
Distance from CoG to end of the link (mm)	95	35.3	163.4	–
Inertia of Link (kgm^2)	0.0445	0.0445	0.0114	–
Friction coefficient of Link (Nms/rad)	1.8	1.5	0.39	–
Lumped mass (kg)	–	–	–	0.3

Table 3.1: *Parameters needed for the Katana 450.*

the rotary axis of actuator 2, 3 and 4, respectively. A pictorial example of the definition of Link 2 is connected to the actuator base 2, which is connected to the ground (defined as Base). The maximum reachable point of the manipulator is approximately 0.6024 m from the actuator 2 base point. The workspace for a manipulator is restricted by three different factors such as structural limits on joint angles, interference between manipulator's links and also limitations of the actuators.

A two axis (two DOFs) non-redundant robot arm based on link 2 and link 4 of the Katana manipulator (rotary joints 2 and 4) was initially taken into account and modelled. Hence, the number of control inputs is equal to the degrees of freedom of the system. In this implementation, rotary joints of links 3, 5 and 6 are locked at zero degree relative angles during the non-redundant implementation. By considering only two axis non-redundant robotic manipulator in the desired system, it simplifies the main point of the trajectory planning problems. Once the method is demonstrated

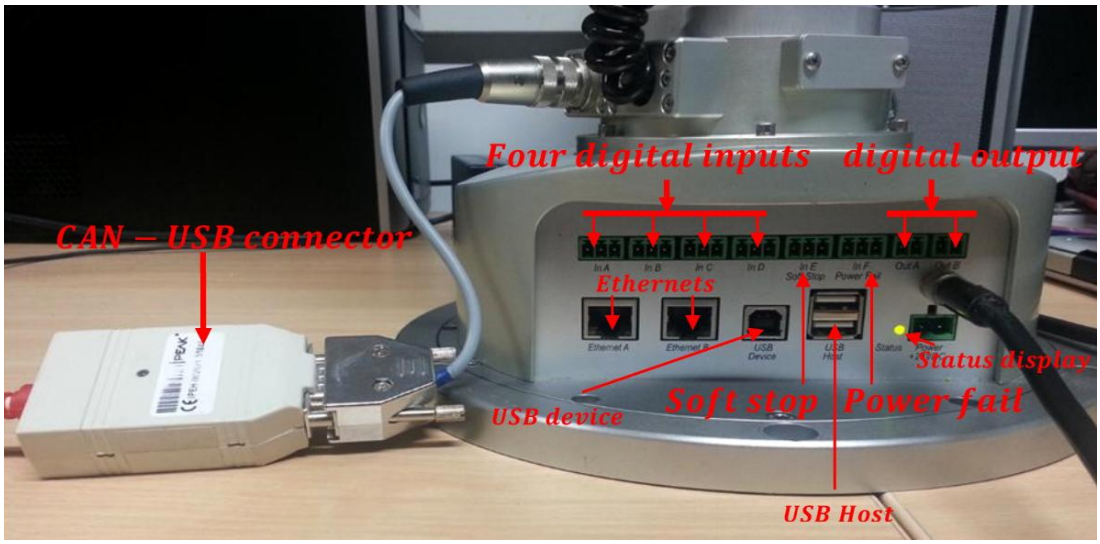

Figure 3-2: *Katana internal control box: view from the front.*

and proven to be feasible on two degrees of freedom non-redundant manipulator, link 2, link 3 and link 4 of the Katana manipulator (rotary joints 2, 3 and 4) are considered for the implementation of 3-link redundant manipulator. In this case, a system has more control inputs than required in order to control a specified desired motion. That is, the robotic manipulator has three DOFs, but the planar system has two degrees of freedom. In this case, the inverse dynamic equations consists of more unknowns than the number of equations [8]. This issue is discussed in more detail in Ch. 6.

In order to control the robotic manipulator, a control box is required. There are two types of control boxes (internal, external types) for the Katana 450 manipulator. An internal model is utilised and directly mounted on the robot's foot as shown in Fig. 3-2. This internal control box has four digital inputs in A-D, soft stop function in E (enables

Links	**Maximum Absolute Angle**	**Relative Angle**
Link 1 (locked)	339°	+/-169.5°
Link 2	**132°**	**+102°/-30°**
Link 3	**245°**	**+/-122.5°**
Link 4	**224°**	**+/-112°**
Link 5 (locked)	336°	+/-168°
Link 6 (locked)	329°	+299.5°/-29.5°

Table 3.2: *Angle limits of Katana 450 6M industrial robotic manipulator.*

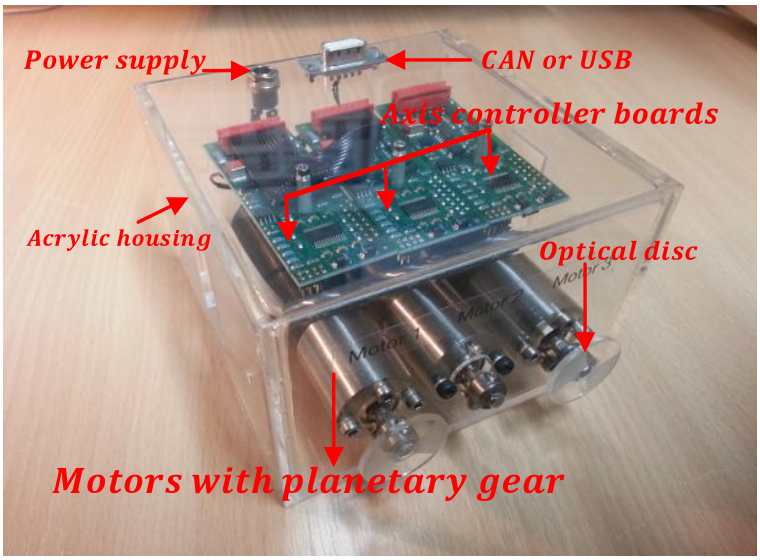

Figure 3-3: *UniKit evaluation board with 3 motors and axes.*

Name	**Link 2**	**Link 3**	**Link 4**
Motor type	Faulhaber-Motor 2657 W 024 CR	Faulhaber-Motor 2642 W 024 CR	Faulhaber-Motor 2657 W 048 CR
Planetary gear	Faulhaber Planetary gear 26/1-3.71	Faulhaber Planetary gear 26/1-3.71	-
Mass (kg)	0.233	0.233	0.182
Gear ratio	371	371	100

Table 3.3: *Katana 450 6M motor and gear specifications.*

the program to be stopped via and external signal), power fail in F (if it is enabled, the robot returns to the home position) and also two digital outputs between output A-B. In addition to these, the USB host ports give the user options to update firmware and to run standalone program memory for the Neuronics control pad. Standalone option provides the user to operate the robot without a PC connected. For this intention, the required program can be exported from Katana4D program and uploaded to the Katana's control board via file transfer protocol (FTP) or to save the data to a USB stick and then inserting into the Katana control box. Another input option is that the USB host also provides an alternative control choice by utilising the Ethernet port. The power supply connection and status display of the robotic manipulator are also located on this control board.

Katana basic data	**Specifications**
Model	Katana 450
Number of axes	6
Carrying load, flange	400g
Reach	517 mm
Workspace volume	0.477 m^3
Drive system	DC motors with incremental system
Tera mass	5.2 kg (without control box and gripper)
Gears	Harmonic Drive
Material	Anodised hig-strength aluminium
Installation position	Vertical upright
Control	PC or Standalone

Katana control box	**Specifications**
Supply voltage	24VDC
Average power consumption	50 W
User inputs	6 digital inputs between input A and F
Input voltage	24V
User outputs	2 digital outputs between output A and B
Output voltage	24V
Main board	PPC MPC5200 400MHz, 32MB Flash, 64MB RAM
Operating system	Embedded GNU/Linux, Kernel 2.4 mit Xenomai

Table 3.4: *Technical specifications of Katana 450 manipulator.*

The required experimental study can be firstly analysed and executed in the Katana 450 evaluation board before it is implemented on the Katana 450 manipulator. The evaluation board provides identical axis controller and joint motors similar to the Katana robotic manipulator. The main intention of using this evaluation board is that it influences, monitors and visualizes the intended process and communication data in the system to prevent the risk of damaging to the Katana's actuators. The UniKit board consists of 3 motors and 3 axis controllers as shown in Fig. 3-3. The hardware supplied with the UniKit consists of a motor with planetary gear and optical disc, one axis controller for per motor, connectors for CAN and USB, power supply connector and extension cables for additional evaluation boards.

For theoretical and experimental studies, an accurate computational model of the Katana 450 robotic manipulator and its motors was constructed by using a software package called DYSIM [8] which utilises the Lagrangian dynamics. To construct the model, DYSIM requires the user to specify the mass, inertia and position of centre of

mass for each link, as well as identifying the location of ground point. All these required parameters for DYSIM are obtained from Neuronics Company [2] and are summarised in Tab 3.1.

The physical constraints of the Katana 450 6M manipulator have to be taken into consideration in order to handle the required task accurately. Maximum and minimum angle limitations of the industrial manipulator are given in Tab. 3.2. In addition to these, the motor and gear specifications are also provided in Tab. 3.3. Notably Tab. 6.4 provides general data in relation to the Katana basic and Katana control box specifications.

3.2 Modelling the Katana 450 6M Industrial Manipulator

In order to get the good performance, it is first essential to develop a good theoretical model of the Katana 450 6M industrial robotic manipulator. The development process involves both accurate dynamic modelling of the Katana 450 robotic manipulator and generating the optimal trajectory control process. Trajectory control process involves calculating the optimal manipulator trajectory for the robotic manipulator. The optimum desired trajectory (fifth order B-spline trajectory) is also converted to a digital specified format (cubic polynomial trajectory) which is required by the AxNI interface Katana 450 robotic manipulator program (AxNI interface program will be discussed in the subsequent sections). The theoretical model of the Katana 450 robotic manipulator is verified firstly in simulation, and then experimentally by utilising the UniKit evaluation board and also Katana 450 industrial robotic system for various trajectories.

3.2.1 Inverse Dynamic Analysis

Lagrangian formulation provides a multi-domain system to be modelled readily. DYSIM utilizes Lagrangian Dynamics in order to generate the equations of motion of the required system. For any required system with M degrees of freedom and with $Z \geq M$ generalised coordinates of motion whereby $q_i, i = 1 \cdots Z$, the Lagrangian equation can be given as [8]:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} + \sum_{j=1}^{Z-M} \lambda_j \frac{\partial f_j}{\partial q_i} = Q_i \quad i = 1 \cdots Z \tag{3.1}$$

The variables in Eq. (3.1) are described in the Nomenclature section. The Lagrangian function L of any system can be defined as the difference between the kinetic and potential energy of the required system and can be written in the following general configuration in terms of the generalized coordinates [8].

$$L = \frac{1}{2}\sum_{i=1}^{Z}\sum_{j=1}^{Z} a_{i,j}\dot{q}_i\dot{q}_j + \sum_{i=1}^{Z} a_{i,0}\dot{q}_i + a_{0,0} \tag{3.2}$$

Due to superfluous coordinates in the system [8], $Z - M$ constraint equations are required, $a_{i,j}$, $a_{i,0}$, $a_{0,0}$ are functions of generalized coordinates and time.

$$f_j = 0, \qquad j = 1 \ldots (Z - M) \tag{3.3}$$

Inserting Eq. (3.2) into Eq. (3.1) and taking the double differentiation of Eq. (3.3), following $2Z - M$ differential algebraic equations can be obtained [8]:

$$\begin{bmatrix} \mathbf{A} & \mathbf{F}^T \\ \mathbf{F} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{q}} \\ \lambda \end{bmatrix} = \begin{bmatrix} \mathbf{D_1} \\ \mathbf{D_2} \end{bmatrix} + \begin{bmatrix} \mathbf{Q} \\ \mathbf{0} \end{bmatrix} \tag{3.4}$$

where $\mathbf{A}$ is an $Z \times Z$ matrix of $a_{i,j}$ functions, $\mathbf{F}$ is the constraint Jacobian matrix, $\mathbf{0}$ is a matrix of zeros, $\mathbf{D_1}$ and $\mathbf{D_2}$ are vectors of $\mathbf{q}$ and $\dot{\mathbf{q}}$. $2Z - M$ unknowns can be obtained by solving Eq. (3.4), namely the second derivatives of generalised coordinates and Lagrange multipliers. In order to obtain a forward dynamic response, second derivatives can be integrated twice. The time history of Lagrangian multipliers will also be calculated automatically, and hence forces of constraints can be determined [8].

In inverse dynamic analysis, in the case of the non-redundant manipulator system, the desired motion is specified in terms of the second derivative of the generalised coordinates [8]. Assuming that the desired motion has $K = M$ degrees of freedom in

the following form:

$$\ddot{\mathbf{y}} = \mathbf{C}_1\ddot{\mathbf{q}} - \mathbf{C}_2 \tag{3.5}$$

In the case of redundant manipulator system, $K \leq M$ degrees of freedom motion are specified in terms of the second derivatives of the K generalized coordinates [8].

$$\ddot{\mathbf{y}} = \mathbf{C_1}\ \ddot{\mathbf{q}} \tag{3.6}$$

where $\mathbf{C_1}$ of dimensions $K \times N$ is the coefficient matrix to specify motion defining coordinates. The control input vector $\mathbf{U}$ of dimension K can be added to the generalised input vector with a coefficient matrix $\mathbf{B}$ specifying the position of the control action. An inverse dynamic model can be formulated in various ways as discussed in [8], but the general formulation can be obtained as follows for a non-redundant manipulator case by moving the unknown control input vector to the left hand side in Eq. (3.4) and using the desired motion as additional constraint equations [8]:

$$\begin{bmatrix} \mathbf{A} & \mathbf{F}^T & \mathbf{B} \\ \mathbf{F} & \mathbf{0} & \mathbf{0} \\ \mathbf{C}_1 & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{q}} \\ \lambda \\ \mathbf{U} \end{bmatrix} = \begin{bmatrix} \mathbf{D}_1 + \mathbf{Q} \\ \mathbf{D}_2 \\ \ddot{\mathbf{y}} \end{bmatrix} \tag{3.7}$$

Eq. (3.7) can be solved for the second derivatives of the generalised coordinates. In order to obtain the motion of the system generalized coordinates can be double integrated. The required control inputs vector $\mathbf{U}$ and the Lagrange multipliers λ will be automatically calculated during the process [8].

3.2.2 The DYSIM User Interface

In order to construct the Katana 450 manipulator model, DYSIM requires that the user specifies the mass, the moments of inertia of each part about its centre of gravity, the co-ordinates of the connection points of each part in relation to the part's centre of gravity as well as specifying friction in each of the joints and also gear ratios of the each motor [3].

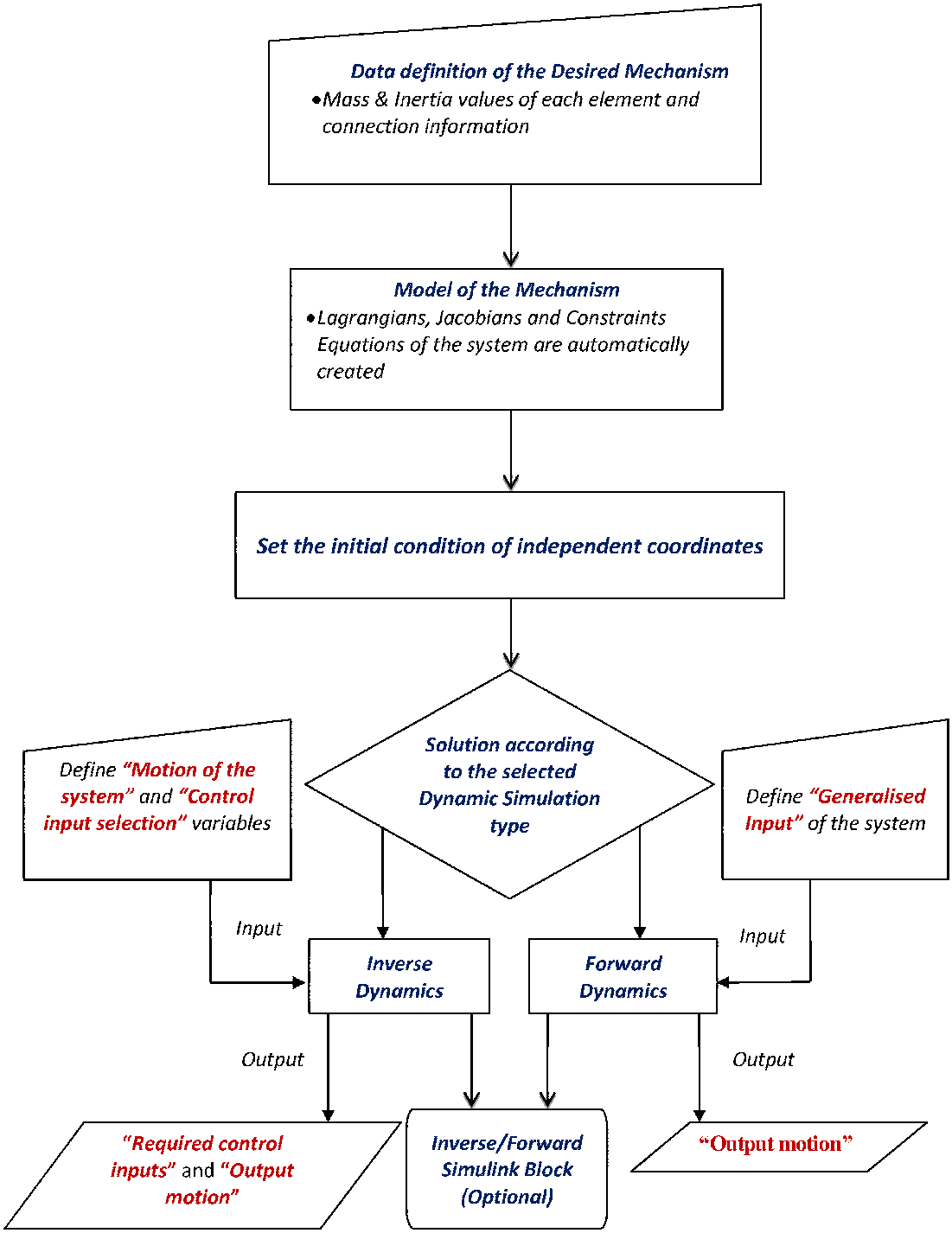

Figure 3-4: *The method of operation of DYSIM dynamic simulation.*

The mass, inertia and link length data of the individual elements of the robotic manipulator as taken from the Neuronics are already summarised in Tab. 3.1 and also motor and gear specifications are given in Tab. 3.3.

After entering the physical and dimensional data, constraint equations, equations of motion and Jacobian expressions are derived automatically by the DYSIM program [3]. DYSIM also provides a list of variables which makeup the required manipulator system. With the manipulator model created, the user also has the ability to add and describe additional *user defined functions* (UDF's) to specify any additional potential and dissipated energy terms.

Following this, initial conditions of the independent variables are required [8]. DYSIM has the ability of performing necessary computations to compute the initial conditions of dependent variables, i.e., positions of the each link point and rates of change of the dependent variables [3]. Finally, the modelling options of the required system should be selected as forwards or inverse dynamic simulation [8]. If an inverse dynamic model is required then the user must identify the *control input* and *motion defining* variables. With the manipulator model now fully described, it can be used from Simulink through the DYSIM toolbox. A flow chart summarising the method of functionality of DYSIM is shown in Fig. 3-4.

The procedure of generating a two-link non-redundant robotic manipulator model of the KATANA 450 6M robot utilising the DYSIM program will be discussed in this section. The procedure of creating 3 link redundant robotic manipulator's model has a similar creation process.

After loading a new project file on the DYSIM program, the *Planar (2D) Mechanism Interface* screen was selected by the user to create the desired model. Utilising this interface screen, the physical and geometric specifications of the manipulator links can be identified in order to generate the required model, as well as interconnection between the robotic manipulator links [3]. The *Planar (2D) Mechanism Interface* screen depicted in Fig. 3-5 can be divided into 5 sections, which have their own particular functions.

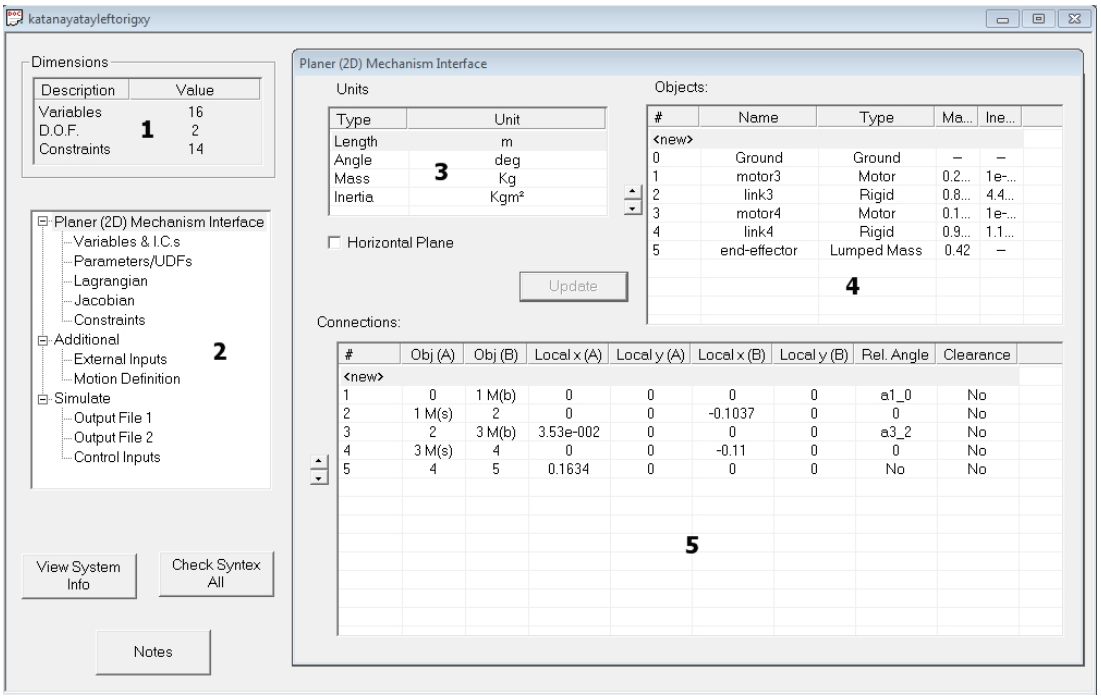

Figure 3-5: *The planar (2D) mechanism interface screen.*

Section 1. In this section of the interface screen, the user cannot enter any parameters. All the values in this section are automatically derived, such as the number of generalised coordinates, DOF, and constraint equations of the robotic manipulator. The number of degrees of freedom can be described as the difference between the number of variables and number of constraints of the generated robotic system. Rigid bar type manipulator's links have three generalised coordinates; the absolute position of the centre of gravity of the manipulator's link in x and y coordinates and angular orientation [3].

Section 2. This section provides access to the different parts of the program during the creation of the mechanism. To access the *Planar (2D) Mechanism Interface* screen, *Planar (2D) Mechanism Interface* must be selected from the sidebar.

Section 3. This section consists of the units of the parameters. The units of the essential parameter values can be described and entered. The position of the required system can be identified as the vertical or horizontal plane. That is, the gravitational effects can be taken into account according to the required system specifications [3].

Section 4. Utilising this part of the interface screen, individual links of the robotic manipulator are being modelled and defined by the user. Typical links, the mass and inertia information have to be given in order to complete the object identification for *rigid bars*. Links can also be defined as point masses or inertias if needed [3]. Point mass manipulator's links present mass values but no inertia value. Similarly point inertia manipulator's links consider inertia values, but not mass value [3]. Each link of the robotic manipulator is defined by an object number by DYSIM, which in a two link non-redundant case as shown in Fig. 3-5, is 0 through to 5. For ease of reference and simplicity, the user can also give a name to each manipulator's link for simplifying of the identification of the robotic links. In all cases, the ground of the body is always presented as *body 0* by DYSIM [3]. Mass, inertia values and also motor specifications for each manipulator's link were provided to the DYSIM program as described in Tab. 3.1 and Tab 3.3, respectively. All the manipulator's links were described as rigid bars. In our system, the first link of the robotic manipulator (define as a number of 2) is connected to the motor gearbox shaft, and the base of the first motor (define as a number of 1) is connected to the ground and so on. The object 5 is described as being the end-effector of the Katana 450 robotic manipulator and is therefore selected as lumped mass with 0.3 kg. DYSIM generates constraint equations, Jacobians, and DOF of the mechanism automatically. There is also an option to define relative angles (in addition to absolute angles) as generalised coordinates.

Section 5. Using this part of the interface screen, each connection of the object is described by the user. In order to specify a connection between manipulator's links, the user has to provide the connection parts in the body fixed (local) coordinate system, which passes through the centre of gravity of each body [3]. An identification number will be assigned to each connection by DYSIM, therefore, in the case of two link non-redundant manipulator, is 1 to 5. DYSIM generates constraint equations, Jacobians, and DOF of the mechanism automatically. There is also an option to define relative angles (in addition to absolute angles) as generalised

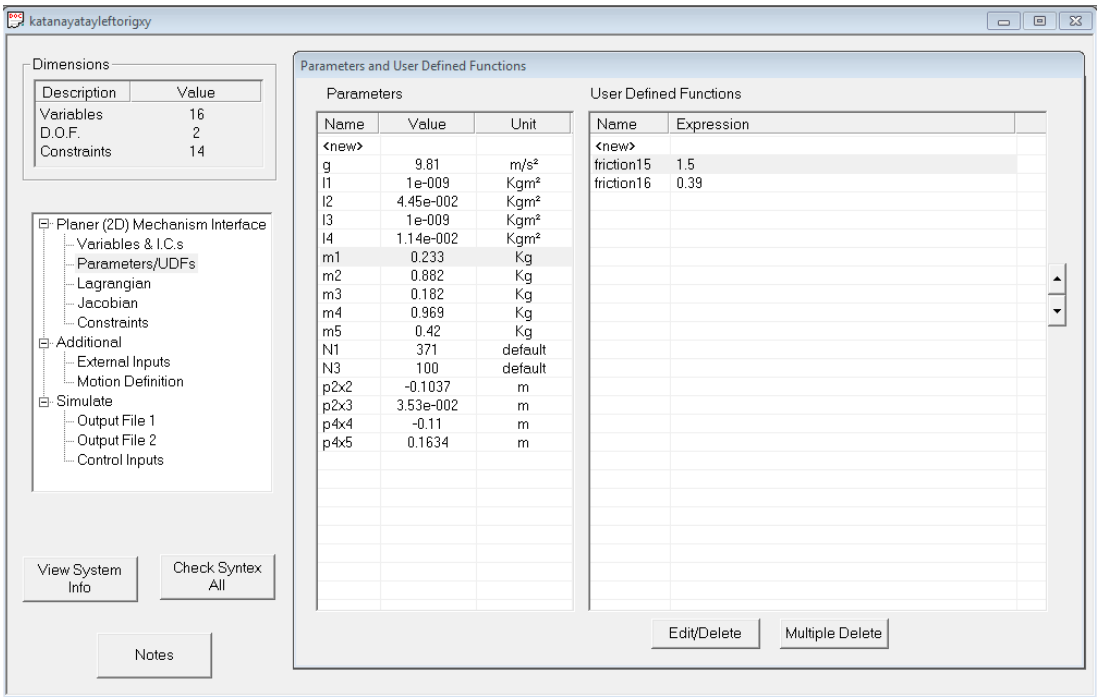

Figure 3-6: *The parameters and user defined functions screen.*

coordinates.

Entering the necessary information via the *2D Planar Mechanism Interface* screen, a series of parameter values defining the positions and masses of the defined manipulators links will automatically be created by DYSIM. These parameter values as well as additional *user defined functions* (UDF's) can be viewed by selecting the *Parameters and User Defined Functions* in the sidebar of the interface screen [3]. A screenshot of this interface screen is demonstrated in Fig. 3-6. The Lagrangian, Jacobian and constraint equations of the required system will be automatically formed by the DYSIM program [3]. Although this section of the screen is not user modifiable, the form of these functions can be seen by clicking the corresponding item on the related section of the interface screen. For illustrative intentions, the 14 constraint equations automatically derived for the KATANA 450 6M robotic manipulator are given in Fig. 3-7. The Lagrangian and Jacobian functions of the system are also presented in the same style. As it is demonstrated in Fig. 3-6, the constraint equations of the required system are given in terms of parametric constants, and variables are expressed as *v1* to *v16*. The initial condition of these generalised variables can be seen by clicking the correspond-

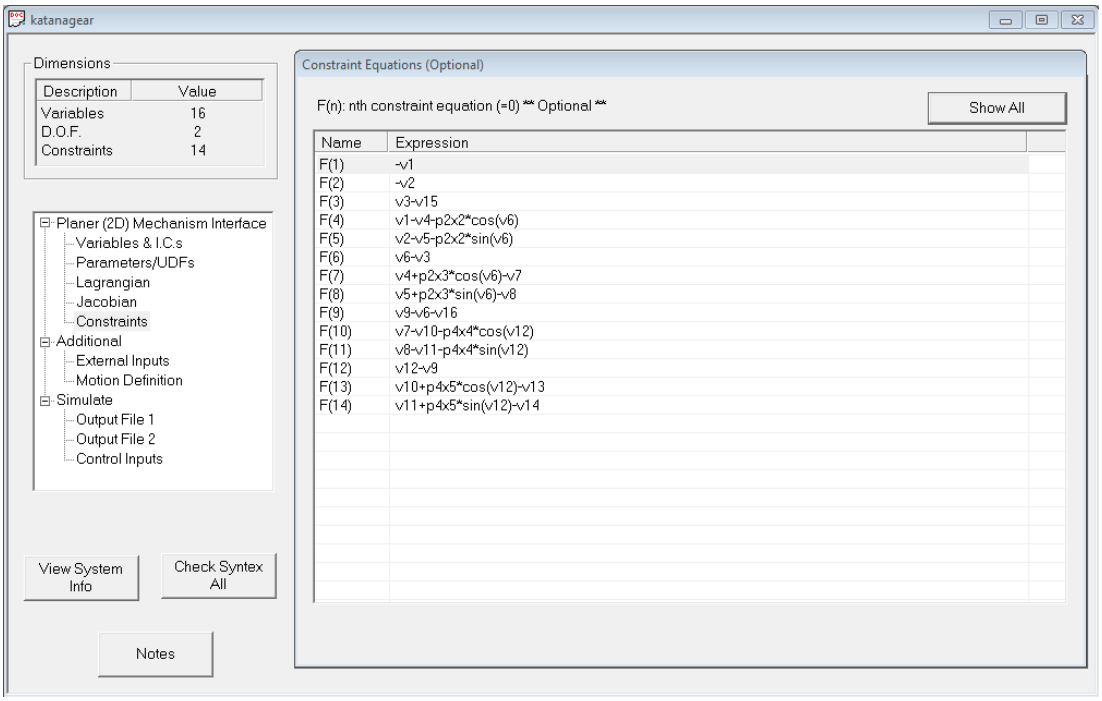

Figure 3-7: *The constraint equation screen.*

ing *Variables & I.Cs* (initial conditions) screen. A screen shot of this interface screen is given in Fig. 3-8. As well as providing a label to each link's variable, the DYSIM program also automatically assigns each link variable a name, x, y and a, for vertical, horizontal and angular displacements, respectively, for each link of the robotic manipulator [3]. Each number demonstrates which manipulator's link the variable relates to. The velocities and the specified unit for each of the variables are given [3].

In order to calculate the variable values, independent variables have to be identified by the user. The number of degrees of freedom of the required system (which is 2 for the non-redundant case) has to be equal the number of independent variables. Therefore, variable *v15*, the relative angular position of the manipulator's link 1 (body 1) and variable *v16*, the relative angular position of the manipulator's link 2 (body 2) were determined as being the independent variable in the required system. If the initial values of the independent variables are given, then click the *Auto Calculate* button, to calculate the initial values of the dependent variables and derivatives. The user can see a schematic view of the desired mechanism at its initial position by clicking the *View* button on the user interface screen. The user can introduce additional potential energy

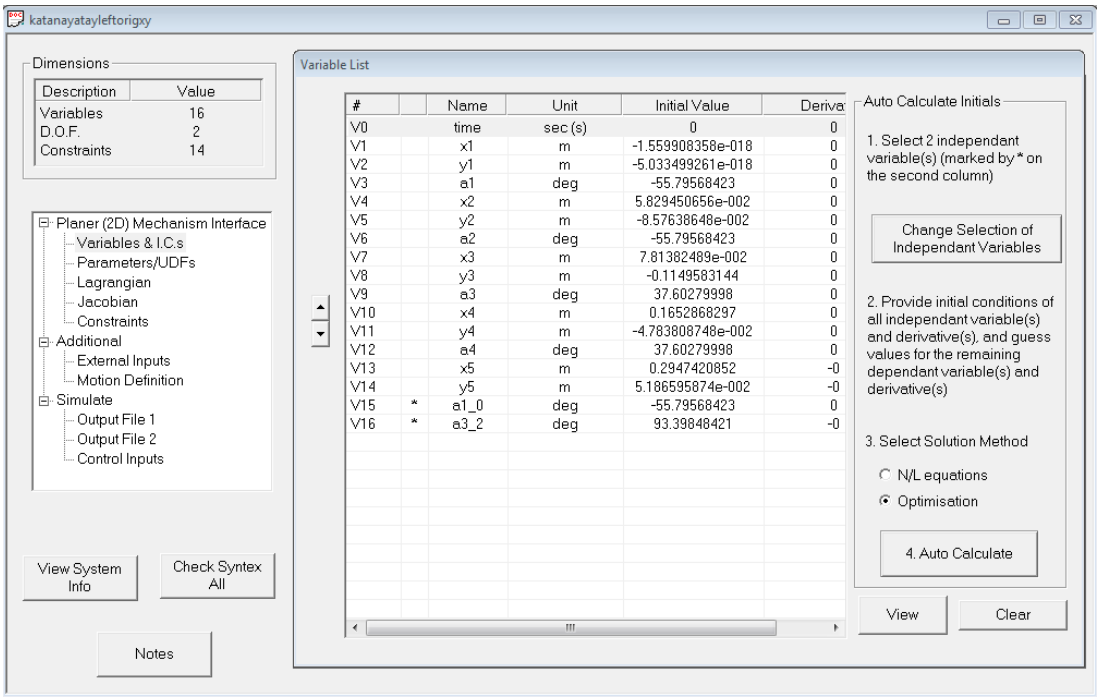

Figure 3-8: *The variables and initial conditions screen.*

or power functions to the system by clicking the *Additional* option in the sidebar [3].

Finally, the required modelling option has to be determined whether forward or inverse dynamic simulation is required. A forward/inverse dynamic modelling of DYSIM is given in Fig. 3-9. This can be performed by clicking the *Simulation* option on the sidebar. A screenshot of this screen is given in Fig. 3-10. Utilising this screen, the user is able to select the type of desired simulation. If forward dynamics is selected, the user must provide the input signals in selected generalized coordinates. The response of the system in generalized coordinates is then computed [3]. If inverse dynamics is desired, the user has to provide the desired output in terms of acceleration profile of selected generalized coordinates. The control inputs in selected generalized coordinates required to achieve this desired output motion is then computed. Lastly, the user must

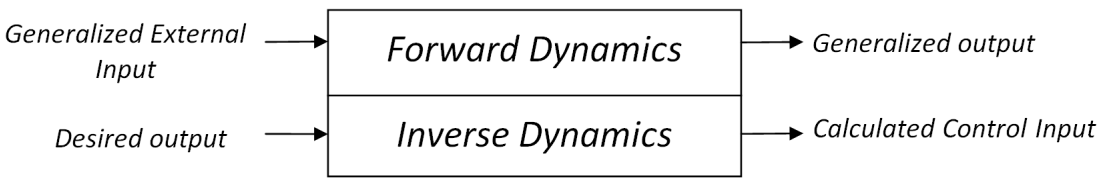

Figure 3-9: *Hybrid forward/inverse dynamic schemes of DYSIM.*

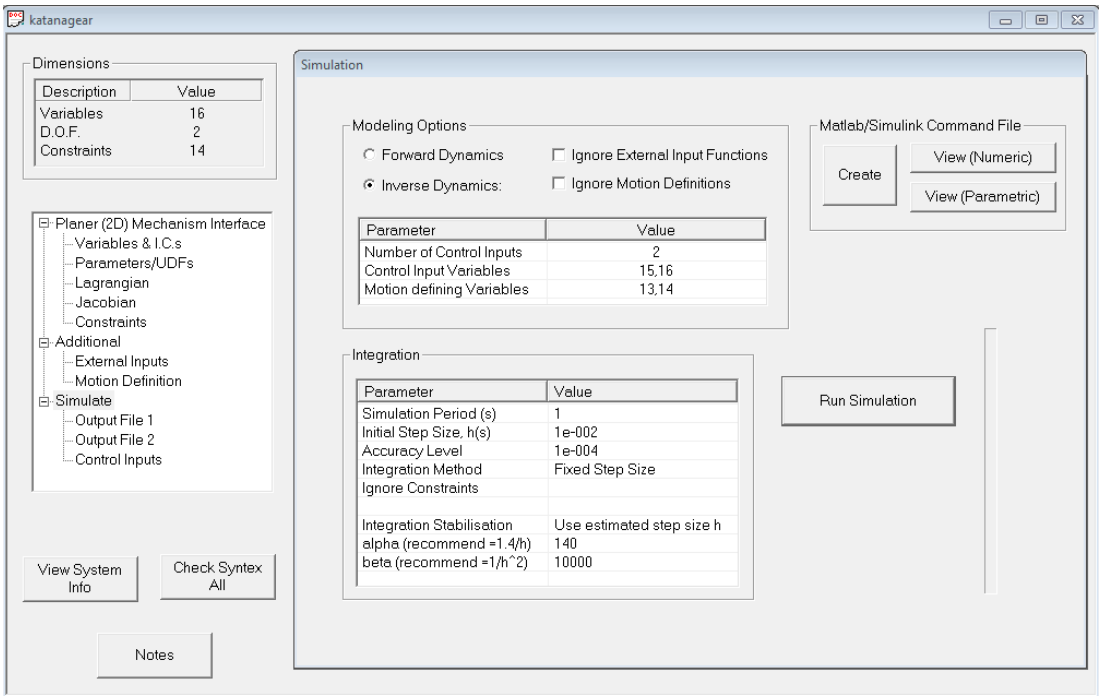

Figure 3-10: *The simulation screen.*

click the *Create* button to export the model to the MATLAB/Simulink environment.

3.3 Hardware Setup and Katana 450 Input Methodology

In addition to modelling the dynamics of the Katana 450 industrial robotic manipulator, it is also essential to set up the hardware as well as identifying the Katana input methodology. To create any movement on the Katana 450 axes, instruction sequence or input data to the axes have to be executed through the Axis Native Interface (AxNI) software. The details of inverse kinematic solution of the manipulator, generating cubic polynomials, procedure of calibrating of each axis, maximum velocity and acceleration checks, AxNI program specification, essential input parameters for Katana axes and experimental procedure and set up will be discussed in this section.

3.3.1 Cubic Polynomial Conversion for Katana 450

To represent the required trajectory for the robotic manipulator, a fifth-order B-spline function was generated in section 4.1.1. In order to implement the optimized trajectory

experimentally, first of all, optimized fifth-order B-spline trajectory has to be converted to cubic polynomials due to input requirements of experimental implementation on Katana. One of the crucial reasons to use cubic splines is that they can be easily fitted to any given trajectory. Hence, the cubic spline conversion is utilised to imitate the optimized B-spline trajectory. After cubic conversion, the created cubic trajectory acts more or less like the original optimized trajectory and the coefficients of the cubic function are utilised as an input into the axis controller for the Katana 450 robotic manipulator [1]. Each piecewise cubic polynomial satisfies continuity up to its second derivative for each polynomial segment. Before the cubic polynomial conversion is implemented, it is assumed that the joint space transformation (t, θ_1, θ_2) from Cartesian coordinate (t, x, y) is already complete. In general, the $S_i(t)$ is a third degree cubic polynomial which can be expressed at a given time, t:

$$S_i(t) = a_{i,1} + a_{i,2}t + a_{i,3}t^2 + a_{i,4}t^3 \tag{3.8}$$

where $a_{i,1}$, $a_{i,2}$, $a_{i,3}$ and $a_{i,4}$ represent the coefficients of the cubic polynomial function. The coefficients can be identified by applying for boundary conditions for each cubic section. As it is known that the third order cubic polynomial satisfies the zero velocity at the beginning and at the end of the desired motion. When a single cubic section ($section_1$) of the cubic polynomial function is taken into account for ease of demonstration, a single cubic section can be expressed based on the notation used in Fig 3-11 as follows:

$$S_{section_1}(t) = a_1 + a_2t + a_3t^2 + a_4t^3 \tag{3.9}$$

where $S_{section1}(t)$ indicates the angular displacement of the joint at the first section on the cubic polynomial curve and a_1, a_2, a_3 and a_4 represent the coefficients of the first cubic polynomial section for t_1 and t_2 time intervals. Between these time intervals, initial and end points of the section are given by S_1 and S_2 and their first derivatives $\dot{S}_1$ and $\dot{S}_2$. A cubic polynomial consists of four coefficients, and will be used to satisfy both position and velocity constraints at the initial and final position of the trajectory.

As is seen from the $section_1$ of the Fig 3-11, the initial (S_1) and final positions (S_2) and velocities ($\dot{S}_1$, $\dot{S}_2$) are the essential boundary conditions for the first cubic $section_1$ of the cubic polynomial function. In order to indicate the starting position of the given trajectory in Fig 3-11, t_1 is accepted as zero second. When the necessary boundary conditions are applied to the first cubic section for t_1 and t_2 time intervals, the initial and final conditions of the cubic ($section_1$) can be shown as follow:

$$S_{section_1}(t_1 = 0) = a_1$$

$$S_{section_1}(t_2) = a_1 + a_2 t_2 + a_3 t_2^2 + a_4 t_2^3$$

$$\frac{d(S_{section_1})}{dt}(t_1 = 0) = a_2$$

$$\frac{d(S_{section_1})}{dt}(t_2) = a_2 + 2a_3 t_2 + 3a_4 t_2^2$$

where $(t_1 = 0)$ and (t_2) are the starting and ending times, respectively. The equations in the above can be shown in a simplified notation as follows:

$$S_{section_1}(t_1 = 0) = S_1$$

$$S_{section_1}(t_2) = S_2$$

$$\frac{d(S_{section_1})}{dt}(t_1 = 0) = \dot{S}_1$$

$$\frac{d(S_{section_1})}{dt}(t_2) = \dot{S}_2$$

S_1 and $\dot{S}_1$ are the specified initial position and velocity, and S_2 and $\dot{S}_2$ are the specified final position and velocity. The coefficients of the cubic $section_1$ can be specified uniquely by using the simplified notation and shown as follows [1]:

$$S_{section_1}(t) = S_1 + \dot{S}_1 t + \left[\frac{3(S_2 - S_1)}{{t_2}^2} - \frac{2\dot{S}_1}{t_2} - \frac{\dot{S}_2}{t_2}\right] t^2 + \left[\frac{2(S_1 - S_2)}{{t_2}^3} + \frac{\dot{S}_1}{{t_2}^2} + \frac{\dot{S}_2}{{t_2}^2}\right] t^3, \quad t_1 \leq t \leq t_2 \tag{3.10}$$

In order to generalise the equation 3.10 for any adjacent cubic sections, notations of S_k and S_{k+1} can be utilised for $1 \leq k \leq n-2$, where n indicates the total number of specified points and the equations can be expressed as follows [1]:

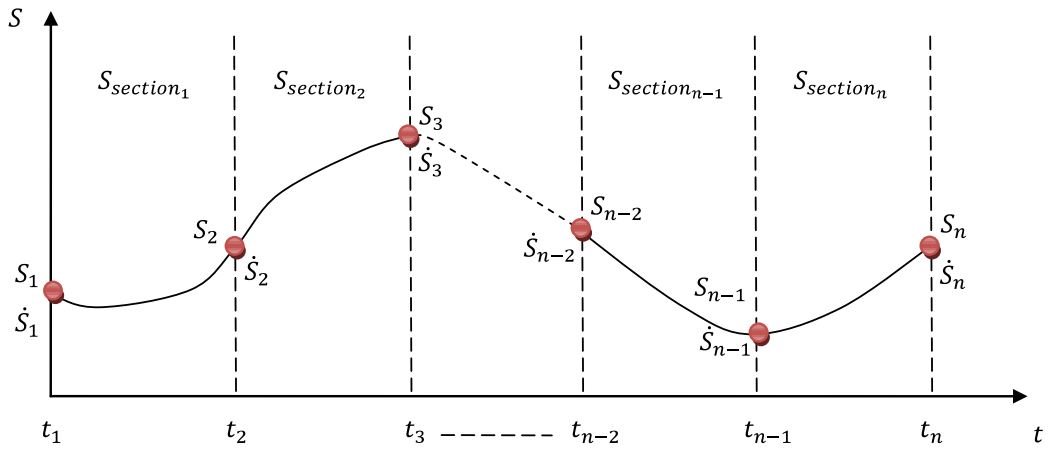

Figure 3-11: *General cubic polynomial representation.*

$$S_{k_{generalized}}(t) = S_k + \dot{S}_k t + \left[\frac{3(S_{k+1} - S_k)}{{t_{k+1}}^2} - \frac{2\dot{S}_k}{t_{k+1}} - \frac{\dot{S}_{k+1}}{t_{k+1}}\right] t^2 + \left[\frac{2(S_k - S_{k+1})}{{t_{k+1}}^3} + \frac{\dot{S}_k}{{t_{k+1}}^2} + \frac{\dot{S}_{k+1}}{{t_{k+1}}^2}\right] t^3 \tag{3.11}$$

$$\begin{aligned} S_{k+1_{generalized}}(t) = \Bigg(& S_{k+1} + \dot{S}_{k+1} t + \left[\frac{3(S_{k+2} - S_{k+1})}{{t_{k+2}}^2} - \frac{2\dot{S}_{k+1}}{t_{k+2}} - \frac{\dot{S}_{k+2}}{t_{k+2}}\right] t^2 \ldots \\ & \ldots + \left[\frac{2(S_{k+1} - S_{k+2})}{{t_{k+2}}^3} + \frac{\dot{S}_{k+1}}{{t_{k+2}}^2} + \frac{\dot{S}_{k+2}}{{t_{k+2}}^2}\right] t^3 \Bigg) \end{aligned} \tag{3.12}$$

A set of adjacent cubic sections for n number of specified points can be generated to define the motion by using the generalised equation 3.11 and 3.12. In order to be able to use the above generalised equations, the user has to know and give all the boundary conditions for $S_1, S_2, S_3, \ldots S_{n-1}, S_n$, and their first derivatives $\dot{S}_1, \dot{S}_2, \dot{S}_3, \ldots \dot{S}_{n-1}, \dot{S}_n$ according to time parameters $t_1, t_2, t_3, \ldots t_{n-1}, t_n$, respectively [1]. However, identifying all of these essential boundary conditions for a given trajectory is a very difficult job for the user. This generalised cubic polynomial equation also defines the motion of the manipulator's joint, however, continuity in the acceleration between each cubic polynomial section is not provided by it. If the continuity can be ensured in the second derivative, in this case, the intermediate velocities can be automatically determined

between two adjacent cubic sections. Normally, a cubic spline should provide the continuity of up to its second derivative and the second derivative of a cubic polynomial section is expressed by:

$$\ddot{S}_{section1}(t) = 2a_3 + 6a_4 t \tag{3.13}$$

As it is given in Eq. 3.11 and Eq. 3.12, these are two adjacent cubic sections which are $S_k(t)$ and $S_{k+1}(t)$. In this case, the acceleration profile of the end of the first cubic section $S_k(t)$, where $t = t_{k+1}$ can be expressed as follows [1]:

$$\ddot{S}_{k_{generalized}}(t_{k+1}) = 2\left[\frac{3(S_{k+1} - S_k)}{{t_{k+1}}^2} - \frac{2\dot{S}_k}{t_{k+1}} - \frac{\dot{S}_{k+1}}{t_{k+1}}\right] + 6\left[\frac{2(S_k - S_{k+1})}{{t_{k+1}}^3} + \frac{\dot{S}_k}{{t_{k+1}}^2} + \frac{\dot{S}_{k+1}}{{t_{k+1}}^2}\right] t_{k+1} \tag{3.14}$$

In the next cubic polynomial section $S_{k+1}(t)$, where $t = 0$, the acceleration profile can be found as follows:

$$\ddot{S}_{k+1_{generalized}}(t = 0) = 2\left[\frac{3(S_{k+2} - S_{k+1})}{{t_{k+2}}^2} - \frac{2\dot{S}_{k+1}}{t_{k+2}} - \frac{\dot{S}_{k+2}}{t_{k+2}}\right] \tag{3.15}$$

After this, the second derivative continuity condition is taken into account as follows:

$$\ddot{S}_{k_{generalized}}(t_{k+1}) = \ddot{S}_{k+1_{generalized}}(t = 0) \tag{3.16}$$

Inserting the Eq. 3.14 and Eq. 3.15 into Eq. 3.16 and simplifiying it, the acceleration continuity condition can be generalized as follows [1]:

$$\begin{aligned} &\Big(t_{k+2}\dot{S}_k + 2(t_{k+2} + t_{k+1})\dot{S}_{k+1} + t_{k+1}\dot{S}_{k+2} \ldots \\ &\ldots = \frac{3}{t_{k+1}t_{k+2}}\left[(t_{k+1})^2(S_{k+2} - S_{k+1}) + (t_{k+2})^2(S_{k+1} - S_k)\right]\Big) \end{aligned} \tag{3.17}$$

where $1 \leq k \leq n-2$. By using the Eq. 3.17 recursively and changing it into matrix form, all the required velocity terms can be computed for the cubic spline [1]. In order to obtain unique solutions for the velocity profile by matrix inversion, the non-square

matrix has to be transformed in to a square matrix. Hence, initial $\dot{S}_1$ and final $\dot{S}_n$ velocities of the equation are set as zero and the intermediate velocity profiles can be determined easily by the given matrix inversion as follows [1]:

$$t_{matrix1} = \begin{bmatrix} 1 & & & & & & \\ t_3 & 2(t_2+t_3) & t_2 & 0 & & & \\ 0 & t_4 & 2(t_3+t_4) & t_3 & 0 & & \\ 0 & 0 & t_5 & 2(t_4+t_5) & t_4 & & \\ & & & . & . & . & \\ & & & & . & . & \\ & & & & & . & \\ & & & & & & 1 \end{bmatrix} \tag{3.18}$$

$$\dot{S}_{matrix} = \begin{bmatrix} \dot{S}_1 \\ \dot{S}_2 \\ \dot{S}_3 \\ \dot{S}_4 \\ . \\ . \\ . \\ \dot{S}_n \end{bmatrix} \tag{3.19}$$

$$S_{accematrix1} = \begin{bmatrix} 0 \\ \frac{3}{t_2 t_3}\Big[(t_2)^2(S_3-S_2)+(t_3)^2(S_2-S_1)\Big] \\ \frac{3}{t_3 t_4}\Big[(t_3)^2(S_4-S_3)+(t_4)^2(S_3-S_2)\Big] \\ . \\ . \\ . \\ \frac{3}{t_{n-1} t_n}\Big[(t_{n-1})^2(S_n-S_{n-1})+(t_n)^2(S_{n-1}-S_{n-2})\Big] \\ 0 \end{bmatrix} \tag{3.20}$$

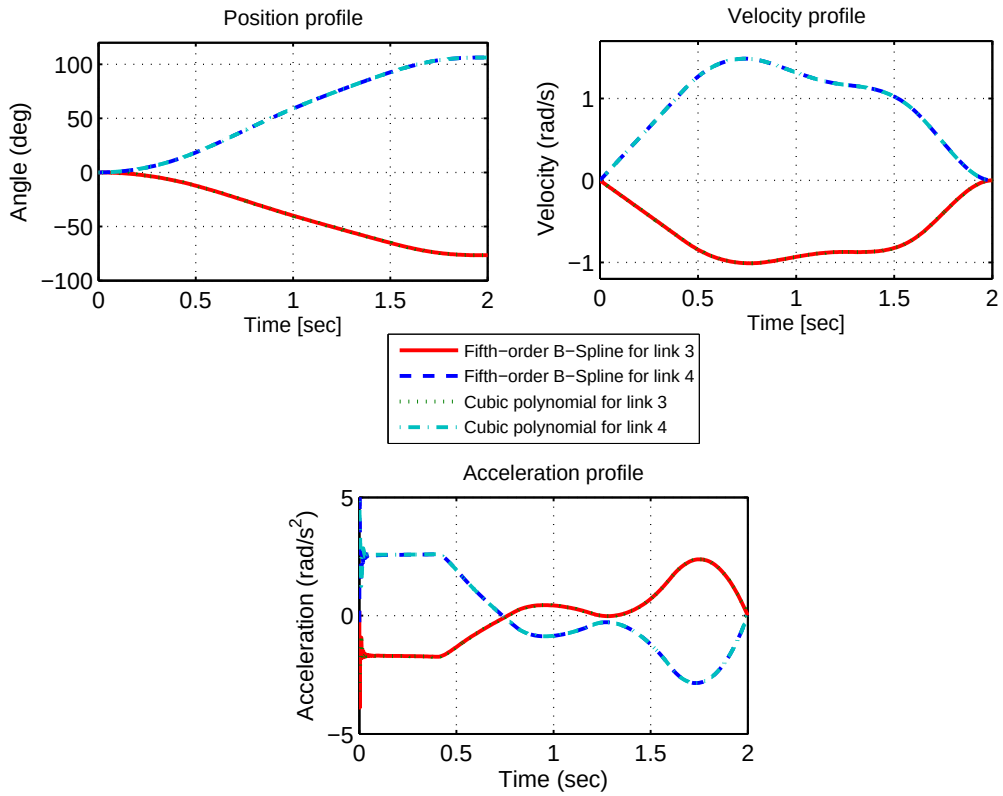

Figure 3-12: *Fifth-order B-spline trajectory with converted third-degree cubic splines.*

$$t_{matrix1} * \dot{S}_{matrix} = S_{accematrix1} \tag{3.21}$$

After the calculation of velocity profiles in cubic spline via Eq. 3.21, the polynomial coefficients for each cubic polynomial section can be easily identified by using Eq. 3.11. A simple flowchart of cubic polynomial calculation is shown in Fig. 3-13. Theoretical position, velocity and acceleration profile comparison between fifth-order B-spline function and converted third-degree cubic polynomials is shown in Fig. 3-12. It is seen from the figure that data generated by a fifth-order uniform B-spline is interpolated by a cubic polynomial which imitates quite well like the original function. Hence, desired trajectory of the system has been reconstructed by cubic polynomial function due to required digital format for the implementation on the Katana 450 robotic manipulator axis.

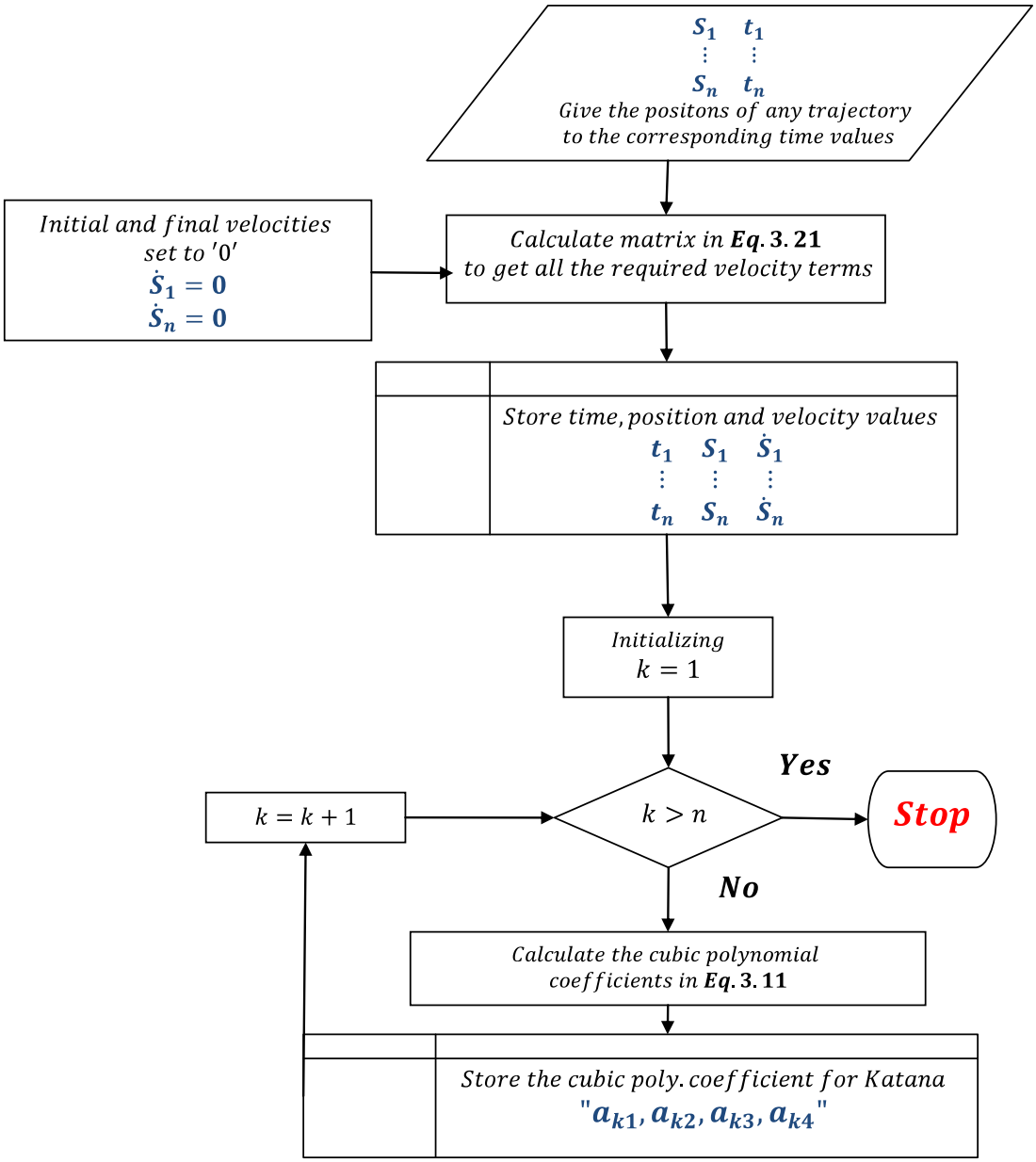

Figure 3-13: *Flowchart of the calculation for the required cubic polynomial coefficients for Katana 450 robotic manipulator [1].*

3.3.2 Katana 450 Robotic Manipulator Axis Calibration

The motors of the Katana 450 6M robotic manipulator utilize their own encoder units, therefore, all the joint motors have to be calibrated before use. The link angles of the robotic manipulator have to be transformed into digital encoder units. The $AxNI$ interface program (will be discussed later in section 3.3.4) enables the user to determine the essential calibration procedure on the robotic manipulator. For each link of the robotic manipulator, the conversion factor is determined based on the number of encoder increments from zero to the upper limit for each unique link angle constraint [2]. For 360° and also absolute angle limit of encoder values for each manipulator's link are given in Tab. 3.5. For the theoretical and experimental studies, links 2 and 4 are utilised in the case of two link non-redundant planar robotic manipulator in Chapter 5 and links 2, 3 and 4 are utilised in the case of three link redundant robotic manipulator in Chapter 6. The other actuators are locked at a relative angle of zero degrees and will not be used during the required motion. The angular constraint for the links in use is demonstrated in Fig. 3-14. The Katana 450 robotic manipulator only uses a positive encoder count in the anti-clockwise direction for link 2 and clockwise for links 3 and 4 [2]. The angular sign conversion utilized in the theory has to be converted to the units utilised by the Katana 450 encoder count as shown in Fig. 3-15. One end of the angle limit of each link will be specified to a zero encoder value, the other end of the angle limit of the link will be a positive maximum encoder count value [1]. The conversion

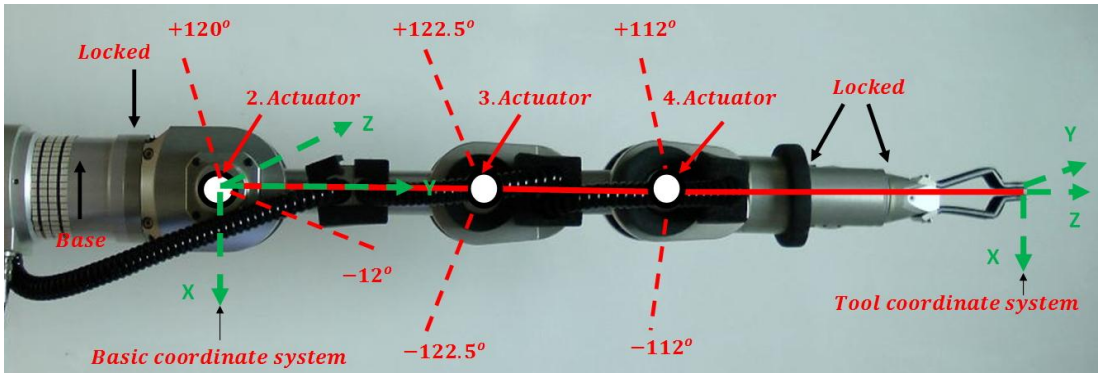

Figure 3-14: *Angle constraints for links 2, 3 and 4 and the other links have been locked with basic and tool coordinate system configurations.*

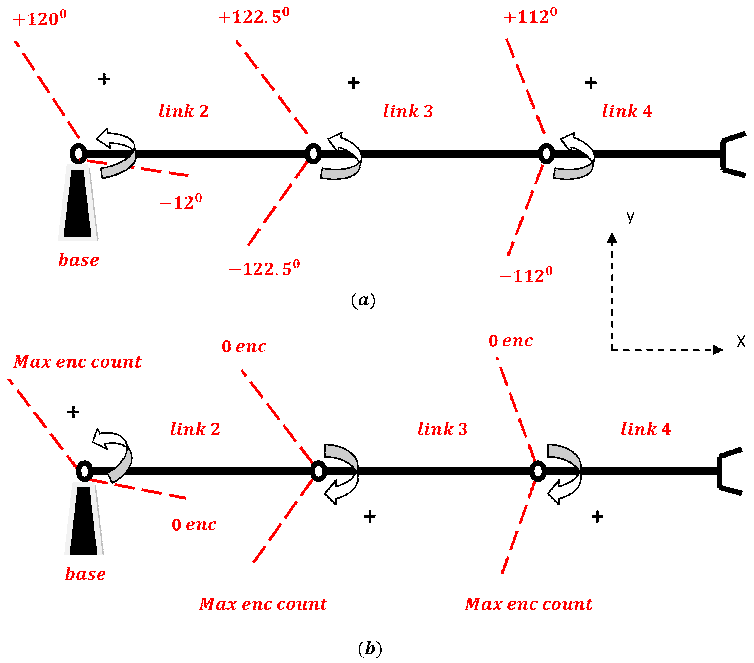

Figure 3-15: *(a) Three-link planar robotic manipulator angular sign conversion with constraints (b) Katana encoder sign conversion for three-link redundant robotic manipulator.*

of joint angles from degrees to encoder count for link 2, 3 and 4 are as follows:

$$\theta_{enc2} = (\theta_{deg2} + 12) * \frac{99280}{360} \tag{3.22}$$

$$\theta_{enc3} = (122.5 - \theta_{deg3}) * \frac{97910}{360} \tag{3.23}$$

$$\theta_{enc4} = (112 - \theta_{deg4}) * \frac{52625}{360} \tag{3.24}$$

In this case, the maximum encoder count under this calibration method can be shown as follows:

$$max(\theta_{enc2}) = (120 + 12) * \frac{99280}{360} \cong 36402 \quad enc \tag{3.25}$$

$$max(\theta_{enc3}) = (122.5 - (-122.5)) * \frac{97910}{360} \cong 66633 \quad enc \tag{3.26}$$

$$max(\theta_{enc4}) = (112 - (-112)) * \frac{52625}{360} \cong 32774 \quad enc \tag{3.27}$$

The modification of the above will guarantee positive values for encoder directions when creating cubic polynomial splines which are needed for the AxNI software of the Katana 450 system. Units in the results and discussion will be presented in a more

Links	Encoder units per 360°	Encoder values for absolute angle limit
Link 1	52200 *enc*	339° / 49155 *enc*
Link 2	**99280** *enc*	**132° / 36402** *enc*
Link 3	**97910** *enc*	**245° / 66633** *enc*
Link 4	**52625** *enc*	**224° / 32774** *enc*
Link 5	52436 *enc*	336° / 48940 *enc*
Link 6	51260 *enc*	329° / 46846 *enc*

Table 3.5: *Katana 450 encoder conversion factor for the links.*

familiar format (degrees, sec, etc.) rather than encoder counts.

3.3.3 Maximum Velocity and Acceleration Checks

The actuators associated with the active links (links 2, 3 and 4) have a maximum velocity profile of 50 $encoder/2.5ms$ and a maximum acceleration profile of 4 $encoder/(2.5ms)^2$ [2]. Because of the variation in gear ratios and calibration procedure for each link of the Katana 450 robotic manipulator, the maximum velocity and acceleration values for each link can be calculated as follows [1]:

For link 2:

$$Vel_{max2} = \frac{50enc}{2.5ms} = 50 * \frac{360}{99280} * \frac{1}{2.5 * 10^{-3}} = 72.52deg/s \tag{3.28}$$

$$Acce_{max2} = \frac{4enc}{(2.5ms)^2} = 4 * \frac{360}{99280} * \frac{1}{(2.5 * 10^{-3})^2} = 2321deg/s^2 \tag{3.29}$$

For link 3:

$$Vel_{max3} = \frac{50enc}{2.5ms} = 50 * \frac{360}{97910} * \frac{1}{2.5 * 10^{-3}} = 73.53deg/s \tag{3.30}$$

$$Acce_{max3} = \frac{4enc}{(2.5ms)^2} = 4 * \frac{360}{97910} * \frac{1}{(2.5 * 10^{-3})^2} = 2353deg/s^2 \tag{3.31}$$

For link 4:

$$Vel_{max4} = \frac{50enc}{2.5ms} = 50 * \frac{360}{52625} * \frac{1}{2.5 * 10^{-3}} = 136.8deg/s \tag{3.32}$$

$$Acce_{max4} = \frac{4enc}{(2.5ms)^2} = 4 * \frac{360}{52625} * \frac{1}{(2.5 * 10^{-3})^2} = 4378deg/s^2 \tag{3.33}$$

To control the maximum velocity and acceleration profiles for a given trajectory, the total duration of the movement can be varied. The following experiments were carried out to establish the fastest operating speed of the Katana 450 robot when actuators 2 and 4 are moving from the initial point ($x_i = 0.4607, \quad y_i = -0.2939$) m and final one ($x_f = 0.4607, \quad y_f = 0.2939$) m respectively with load mass of 0.3 kg. The maximum velocity input into the Katana 450 robotic manipulator's axes was intentionally adjusted in order to exceed its actuator capability and the results are shown in Fig. 3-16. As it is seen from the Fig. 3-16(a), actuator 2 of the Katana manipulator reaches the manufacturer's maximum allowable velocity limit profile at this duration of motion and eventually violates the actuator capability in the Katana manipulator. In this system, when the velocity violation occurs in the actuator 2, output current of the actuator 2 will increase in the system as shown in Fig. 3-16(b) due to the integral term in a PID controller. That is, the integral term is the sum of the instantaneous

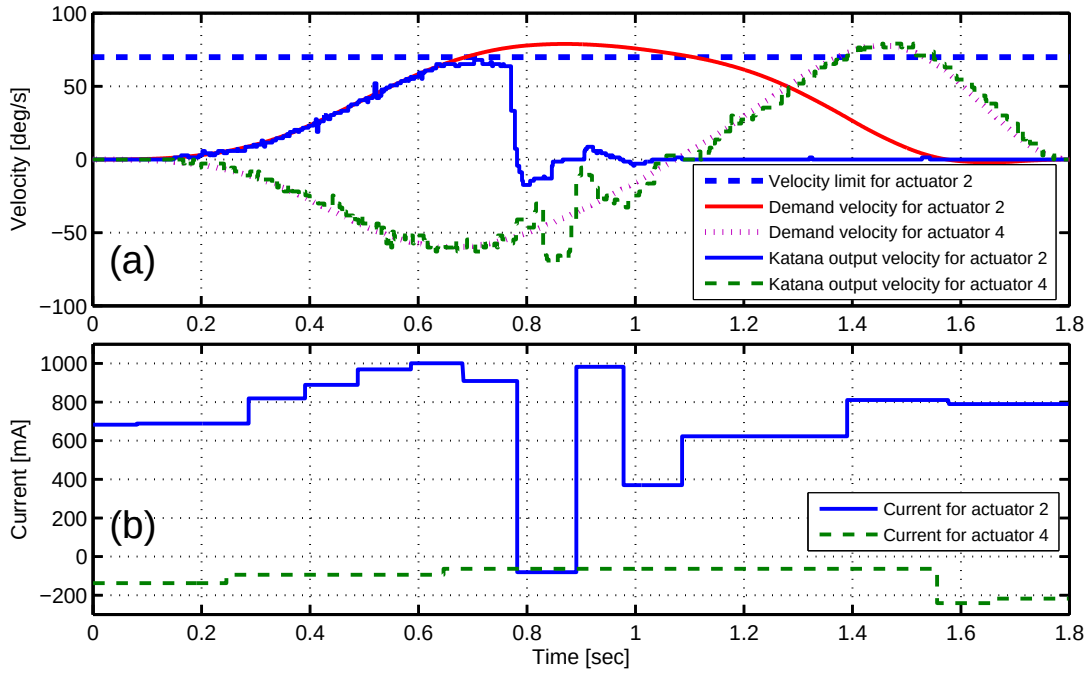

Figure 3-16: *Velocity limits for straight line trajectory (a) The saturation of velocity limits, (b) The current outputs of the Katana during the motion.*

error over time and it gives accumulated error and then this error multiplied by the

integral gain and added to the controller output. Therefore, the controller accepts this error as a collision occurred (due to saturation of integral term in PID) on the link 2 and it stops the actuator 2 movement. Figure. 3-16(b) shows the corresponding motor currents. Therefore, the trajectory with duration of motion of 2 seconds was simulated and calculated as the fastest time possible for the 2 link non-redundant manipulator based on actuator 2 and 4 and it does not exceed its maximum velocity limit for a given trajectory.

3.3.4 Axis Native Interface Program (AxNI)

The Axis Native Interface ($AxNI$) program acts as a main communication connection interface program between the user and the test rig. This interface program is an open source graphical user interface (GUI) software which is written in the python language and it allows the user to do the following [2]:

1. The communication between the hardware and the user can be initiated.
2. The required axis to be controlled can be selected.
3. The axis encoders (*positions*) and the control specifications (such as max current, speed etc.) can be inputted.
4. All the encoder units of each axis can be calibrated.
5. By setting the target position, point-to-point (PTP) motion can be performed via this interface software. In order to carry out more complicated trajectories, the required trajectories have to be converted to a set of third order cubic polynomials manually off-line.

Figure. 3-17 shows AxNI View interface screen after a successful connection via a CAN bus to the axes with the runtime data on the Katana 450 robotic manipulators [2]. The left section of the window indicates generic settings for the experiment. As previously mentioned, the CAN Bus connect section of the window enables engagement of the AxNI to a CAN device. If the successful connection is made, the status boxes on the

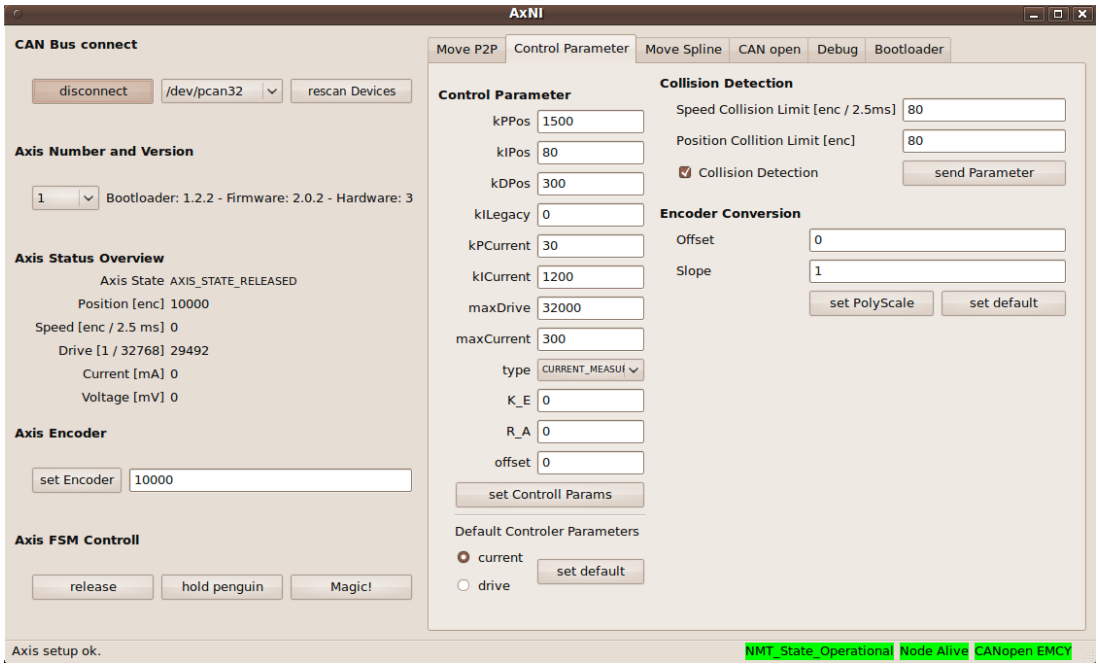

Figure 3-17: *Katana AxNI interface program view after a successful connection [2].*

bottom right of the window turn from red to green [2].

All of the required axes can be selected in the Axis Number and Version box. Normally, the AxNI view allows controlling one axis a time. However, for the 2 link non-redundant and 3 link redundant implementations, all the actuators have to be derived simultaneously. Therefore, the AxNI interface program was modified to implement multiple actuator movements simultaneously.

The Axis Status Overview indicates the current real time data such as position in encoder, speed, drive, current and voltage of the selected axis. In the Axis Encoder section, the encoder value of each axis can be set, and this section is utilised for the calibration of the Katana 450 manipulator's axis [2]. And also, the Axis FSM Control section enables switching to release or hold of each manipulator's axis and executes the selected cubic trajectory by the Magic button.

There is another important section which needs to be taken into account seriously to prevent considerable damage on the Katana 450 manipulator's actuators. The Control Parameter box has to be set according to the each axis capable limitations [2]. When the required axis is selected on the left side of the window, the Control Parameter box

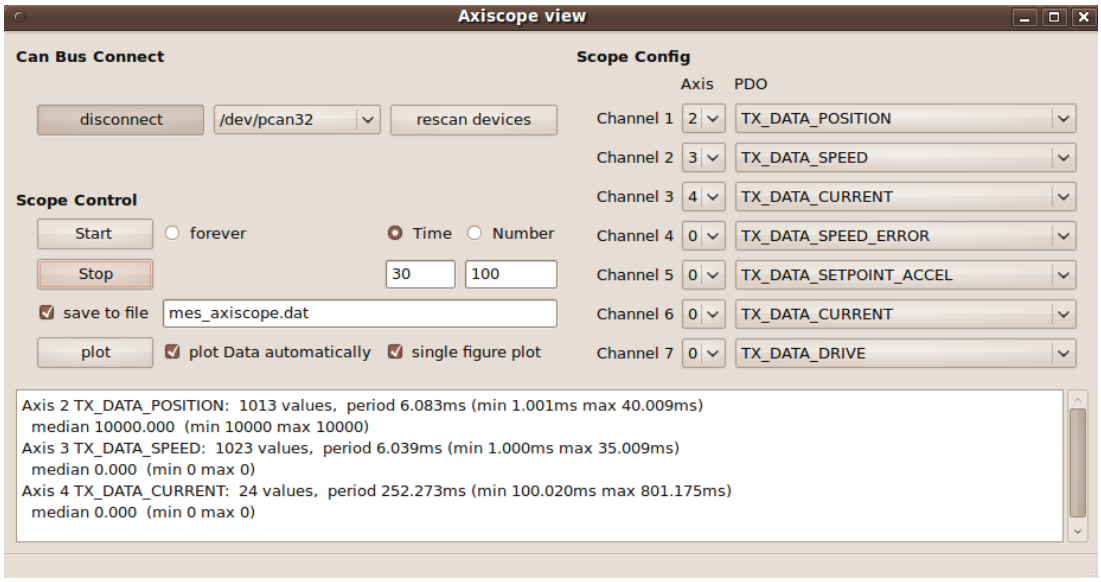

Figure 3-18: *The tool axis scope view*

indicates the corresponding current selected axis control parameters. However, it is recommended by the supplier to utilise the default controller parameters only. The modifications can only be done if one knows exactly what each button is doing when configuring the axis controller parameters for the required motion [2]. Other buttons of the window are not related to the multiple axis movement, therefore these buttons of the window are not mentioned here.

To record and visualize real time experimental data of the manipulator's axis controller, the Axis Native Interface Scope program was utilised to choose the operating axis and save the real time data such as time, angular encoder position, encoder velocity, current, voltage etc., however, it cannot record acceleration data [2]. The corresponding screenshot is given in Fig. 3-18. In order to initiate the program, the connection has to be made to the selected CAN device. Following this, the Scope Config box is used to set the each manipulator's axis and the runtime data tag for as many channels as you want to record. This program provides the user to record up to a total 7 different types of encoder data at any given time [2]. To start to save data, the start button needs to be clicked and to end it by clicking stop when the required motion is finished.

3.3.5 Essential Input Parameters for Katana 450 Robotic Manipulator Implementation

As has been discussed in the Katana 450 calibration section (3.3.2), the Katana robotic manipulator takes into account its own encoder units [2]. Another crucial parameter which is utilized by the Katana 450 robotic manipulator is the time variable. Instead of denoting trajectory in the second unit, the Katana robotic manipulator takes into account the time variable in steps of 10 ms, the range of is encoders. By this way, velocity and acceleration profiles are expressed in $enc/10$ ms, and $enc/10$ ms^2, respectively [1]. All the coefficients of the generated cubic polynomial (a_1, a_2, a_3, a_4) function have to be consistent with the units utilised by the Katana 450 robotic manipulator. Therefore, all the coefficients of the generated cubic polynomials have to be multiplied by 2^0, 2^6, 2^{10}, 2^{15}, respectively and rounded off to the nearest integer (refer to Katana 450 user manual). The basic procedure of the multiplication is demonstrated as follows [1]:

$$\begin{aligned} E_1 &= a_1 * 2^0 = a_1 * 1 \\ E_2 &= a_2 * 2^6 = a_2 * 64 \\ E_3 &= a_3 * 2^{10} = a_3 * 1024 \\ E_4 &= a_4 * 2^{15} = a_4 * 32768 \end{aligned} \tag{3.34}$$

After this transformation, coefficients of each cubic polynomial will be inputted into the Katana 450 main program as in the following format [1]:

$$[Target, time, E_1, E_2, E_3, E_4] \tag{3.35}$$

All of these converted parameters with the required format are stored as text files. The process of the required encoder conversion of the coefficient of the cubic polynomials is given in the flowchart in Fig 3-19.

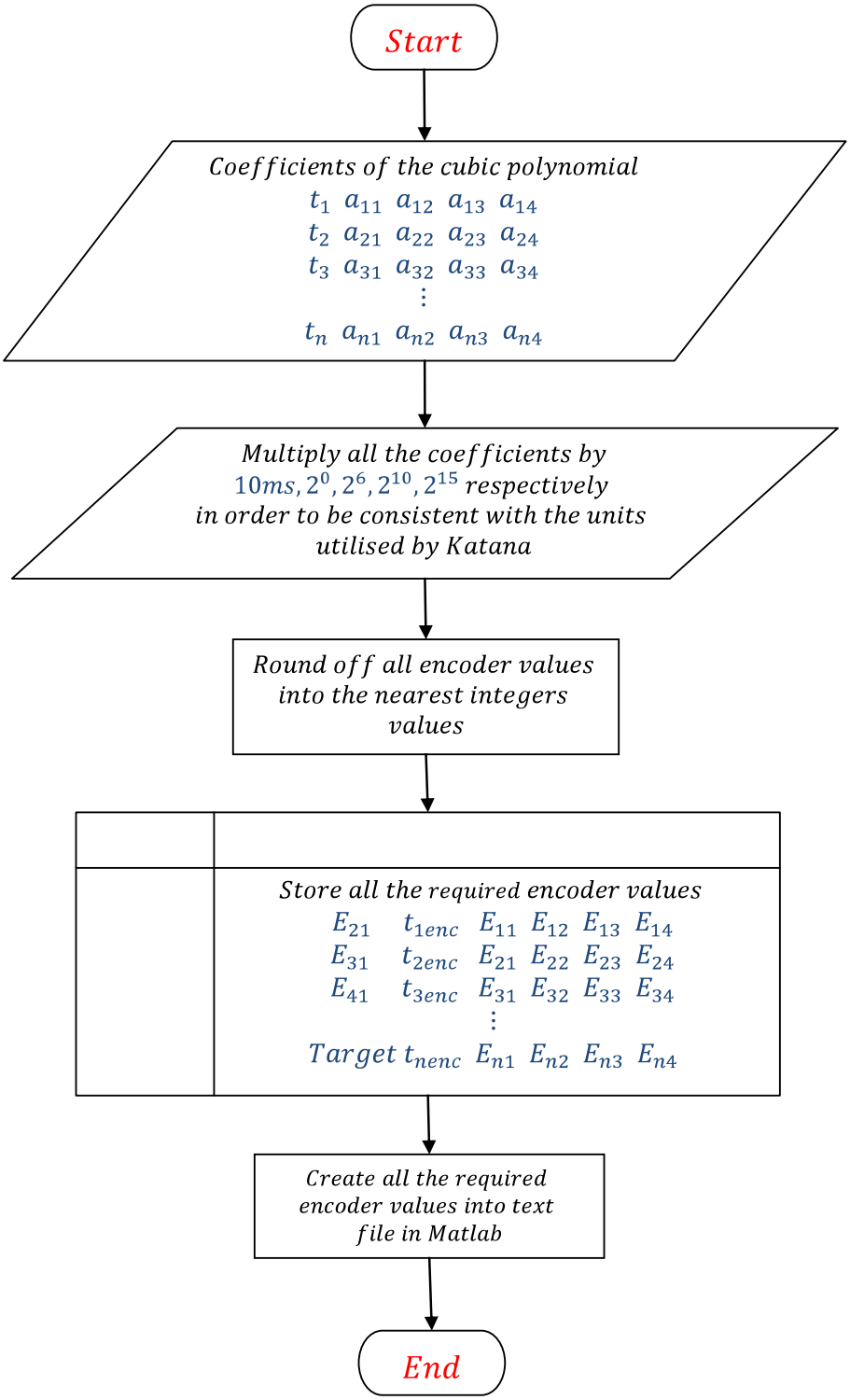

Figure 3-19: *Flowchart for the implementation of the modified input parameters for Katana 450 robotic manipulator.*

3.3.6 Experimental Procedure for Testing Any Trajectories

To implement any trajectory based on cubic polynomial movements on the UniKit evaluation board or Katana 450 robotic manipulator, the general experimental process is as follows:

1. The required trajectory in Cartesian (x, y) (need to be converted into the joint space) or Joint (θ_1, θ_2) coordinates with assigned time t were created by the trajectory optimization program.

2. The main cubic conversion MATLAB program of the Katana 450 robotic manipulator was then implemented to create a set of cubic polynomials for each manipulator's link. In this conversion program, SI units will automatically be transformed to digital units which is utilised by the Katana 450 robotic manipulator. Outputs of the program were in the form of text files.

3. The CAN-USB connection was then linked to the UniKit evaluation board of the Katana 450 robotic manipulator which was controlled by the AxNI interface program in the computer work station as shown in Fig. 3-21.

4. All of the required data in a text file then was utilized as an input into the AxNI interface program.

5. When the program is executed, the position of each actuator moves to its initial encoder positions.

6. To record the desired data in real time, the AxNI interface scope program was executed alongside the AxNI interface program during the required motion.

7. All of the required encoder data can then be taken automatically from the AxNI Scope.

A schematic diagram of Katana 450 experimental setup is also shown in Fig. 3-21.

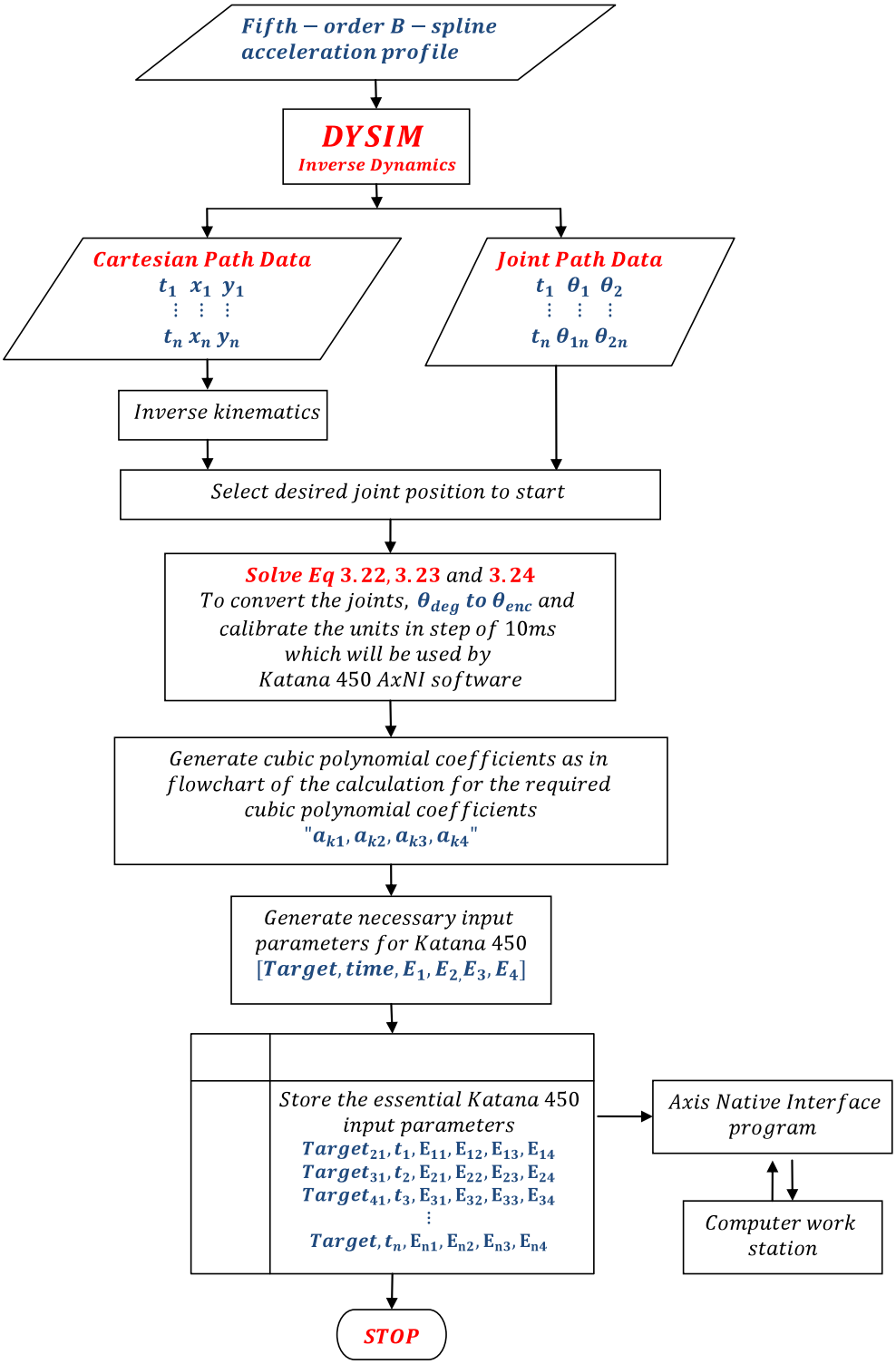

Figure 3-20: *Flowchart for the main program structure of the implementation of Katana 450 robotic manipulator.*

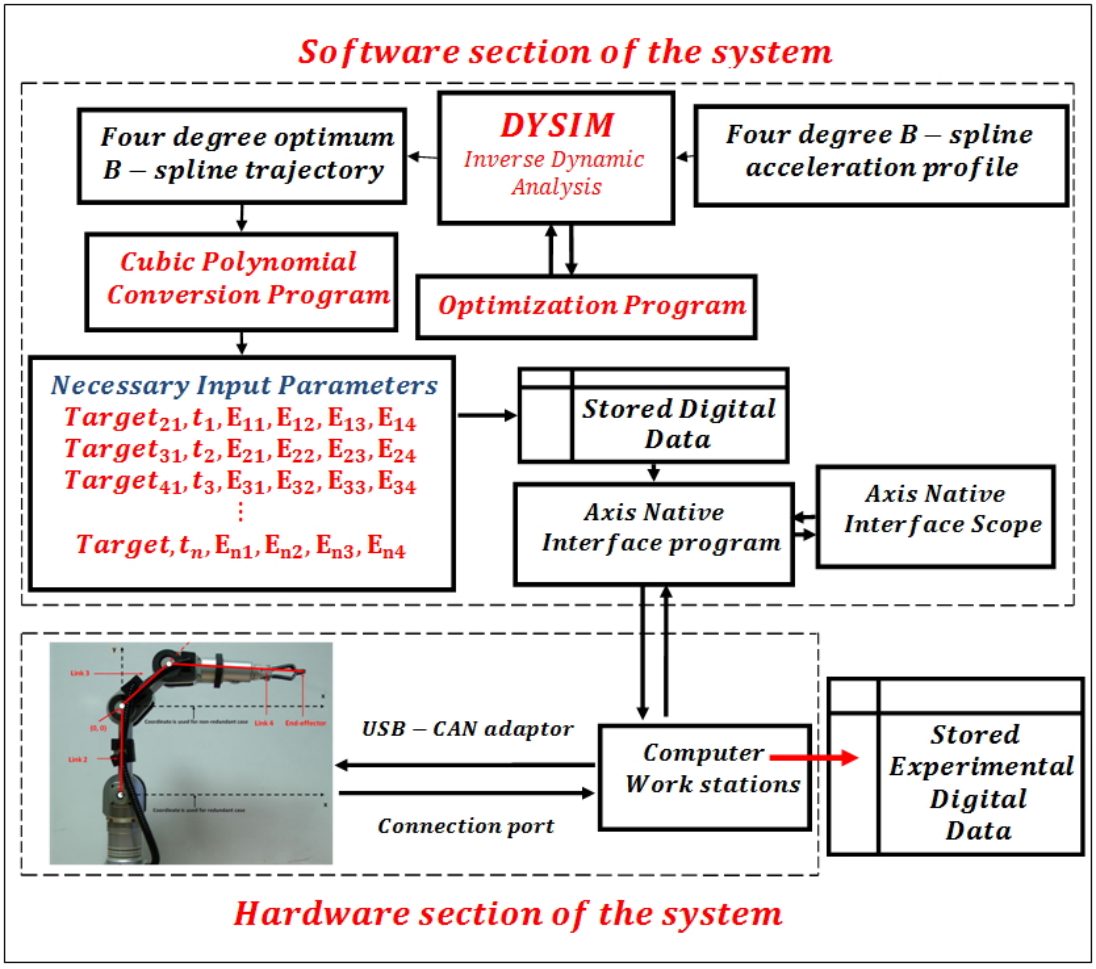

Figure 3-21: *Schematic diagram of Katana 450 experimental setup.*

3.4 Concluding Remarks

In this chapter, work was implemented to develop a model for the Katana 450 industrial robotic manipulator system. All of the physical specifications are taken into account as provided by Neuronics AG Company. A numerical planar model of the Katana 450 industrial robotic manipulator was created by utilising Lagrangian equations and the software Dysim. The robotic mechanism was examined utilising an inverse dynamics procedure.

To implement the proposed methods on the actual robotic manipulator, a stable and precise computational model of the Katana 450 6M industrial robotic manipulator has been created that incorporates the three main degrees of freedom of the Katana robotic manipulator on the basis of links 2, 3 and 4. The created model of the Katana

450 robotic manipulator allows the user to work in both joint and Cartesian coordinate systems. It is based on Lagrangian dynamics, which describe the required system in terms of energy and provide a multi-domain system for modelling. For trajectory optimization implementations, the inverse dynamic model of the Katana robotic manipulators, created by using the DYSIM software, is used. The built-in Planar (2D) Interface of the program is utilised for dynamic modelling of a non-redundant, redundant and also hyper-redundant robotic manipulators. The models can be utilised for either forward or inverse dynamic implementations.

The models developed by Dysim were moved into the MATLAB/Simulink environment, where they could be utilised more effectively to perform and create the various simulation tasks. All of the theoretical data was observed from the MATLAB/Simulink environment, and then this optimized data as sent to the main Axis Native Interface program to drive the actuators of the Katana robotic manipulator. The Axis Native Interface Scope program was then utilised to record and visualize real time optimized experimental data of the Katana robotic manipulator.

An off-line trajectory planning program was also developed successfully for a two-link non-redundant and three link redundant industrial robotic manipulator based on the Katana 450 robot specifications. In order to ensure required digital format for implementation on the Katana 450 axis, various programs were written in MATLAB to transform the time dependent trajectory of the robotic manipulator into a set of cubic polynomial input data that ensures continuity for position, velocity and acceleration at the switching point.

CHAPTER 4

THE PROCEDURE OF OPTIMUM TRAJECTORY PLANNING FOR ROBOTIC MANIPULATORS

In Ch. 3, a model of the Katana 450 robotic manipulator was investigated and developed and also methods utilised for computing the required input parameters for Katana 450 implementation was described, followed by the general experimental setup and procedure.

This chapter deals with the procedure behind the trajectory optimization method for various types of robotic manipulators. Some background on achieving minimum energy consumption trajectories for a point-to-point motion under kinematic and dynamic constraints is also given. To derive the manipulator trajectory, a multi-parametric trajectory optimization method is utilised. The actuator torque has been considered for the formulation of the cost function for the simulation study. In order to compute the cost function for the experimental work, the measured current in each of the actuators is taken into account. Compared with the other trajectory optimization techniques, the proposed method allows the kinematic and dynamic constraints to be included in the cost function reducing the complexity and computational effort of the trajectory optimization algorithm.

A fifth order B-spline function is used to define trajectories for the simulation study. To run the optimized trajectory experimentally, the resultant trajectory was converted

to cubic polynomials in order to meet the Katana input requirements. Both the dynamic analysis of the mechanism and also the trajectory optimization are based on the inverse dynamic analysis. This proposed optimization method will be implemented in various theoretical and experimental studies in Ch. 5 and Ch. 6.

The proposed optimization method has the following advantages:

- Desired trajectory is based on continuous functions.
- Trajectory optimization algorithm is computationally efficient as kinematic and dynamic constraints are included in the cost function to prevent running the inverse dynamic model when all constraints are not satisfied.
- The proposed optimization method can be implemented for various types of robots including redundant/hyper-redundant and parallel robots.

4.1 Path Optimization Problems

Path optimization problems can be divided into following components:

- Selection of parametric path function
- Selection of cost function
- Selection of optimization technique
- Selection of system constraints
- Selection of path coordinates

These are discussed below.

4.1.1 Defining Trajectory by means of Fifth Order B-Spline Function

Selection of the parametric path function is the first step of the trajectory optimization technique. The resultant trajectories in both joint and Cartesian space schemes can be represented in a number of ways. The three most common functions to describe the manipulator trajectories are:

1. Multi-degree polynomial functions
2. Exponential functions
3. B-spline functions

The parameters to be selected should ensure the following features [9]:

- It is crucial to have as a small number of parameters as possible without restricting the motion space for efficiency and convergence of the trajectory optimization process.
- The parametric path function has to be at least twice differentiable in order to ensure a smooth and continuous acceleration profile for the inverse dynamic solution.
- The parameters should provide numerically stable features, that is, they should not be very sensitive to small variations of parameter values to prevent building up of rounding errors. High order polynomial functions have this problem.
- The parameters should have some physical meaning in order to set limits of values as well as initial values for the trajectory optimization. For example, coefficients of polynomial functions do not have physical meanings and it is difficult to set limits.

In this thesis, a uniform fifth-order B-spline function was utilised to describe the required Cartesian and also joint motion trajectories, because of its simplicity and computational efficiency. Various degrees of B-spline functions can be created according to design requirements. The B-spline function has been utilised in many different fields such as computer-aided design, computer graphics, numerical analysis and so on [47]. The utilization of the B-spline function in trajectory planning algorithm has been a very crucial method because the created B-spline manipulator trajectories give continuous values of acceleration profile. The B-spline function utilised in this study is based on piecewise approximations of polynomial functions in order to achieve local control and the fifth-order B-spline function consists of five segments, and each segment is a

polynomial function with a maximum degree of four (for details see [47]). Three steps are required to define a uniform fifth-order B-spline curve at a given time, t [9]:

1. Find the section in which t lies as follows:

$$n_{section} = int(\frac{t}{T/5}) + 1 \tag{4.1}$$

where T is the duration of the signal.

2. Compute the fifth order basis functions as follows [9]:

$$\mathbf{b}(x) = \begin{bmatrix} x^4 & x^3 & x^2 & x & 1 \end{bmatrix} (\frac{1}{4!}) \begin{bmatrix} 1 & -4 & 6 & -4 & 1 \\ -4 & 12 & -12 & 4 & 0 \\ 6 & -6 & -6 & 6 & 0 \\ -4 & -12 & 12 & 4 & 0 \\ 1 & 11 & 11 & 1 & 0 \end{bmatrix} \tag{4.2}$$

where

$$x = mod(\frac{t}{T/5}), \quad x \in [0, 1] \tag{4.3}$$

The numbers in the matrix represent the coefficients of the fifth-order uniform B-spline curves in each section of the B-spline trajectory.

3. Construct the uniform B-spline function by using 9 control points, $(r_1 \cdots r_9)$ as follows [9]:

$$B_{spline}(t) = \sum_{i=0}^{4} b_{i+1}(x) r_{n_{section}+i} \tag{4.4}$$

A fifth-order B-spline function with 5 sections requires nine control points, r_1 to r_9. Three control points (r_2, r_3 and r_4) were utilised to satisfy the initial conditions (position, its first and second derivatives) and the other three control points (r_7, r_8 and r_9) were used to satisfy the end conditions (position, its first and second derivatives). The remaining three free control points (r_1, r_5 and r_6) are optimized by the optimization algorithm [9].

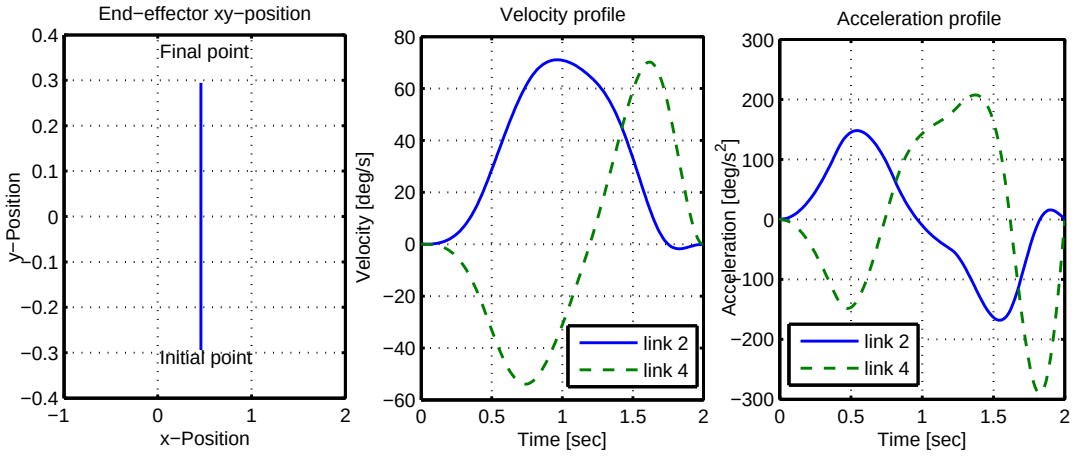

Figure 4-1: *The position, velocity and acceleration profiles for a 2 link robotic manipulator for a 2-dimensional fifth-order B-Spline curve.*

During the generating process of free optimization parameters to be optimized, the parameters of r_1, r_5 and r_6 were selected randomly. However, any other free optimization parameter can also be used and selected as initial parameters for the trajectory optimization algorithm. There is no restriction in the selection process of the free parameters. Therefore, this notation will be utilised in all the theoretical and experimental studies. The position, velocity and acceleration profiles of fifth-order B-spline function are shown in Fig. 4-1. As it is seen from the Fig. 4-1, the values of first and second derivatives of the position profile are zero at the initial and final position. As mentioned in subsection 3.3.1, the desired fifth-order B-spline trajectory will be converted to multiple cubic polynomials in order to ensure the required input parameters for Katana 450 axes in the experimental implementation.

4.1.2 Cost Function

Trajectory optimization can be described as the procedure of finding the minimum or maximum values of a cost function, also called the objective function, evaluation function etc. If the required cost function is something to be minimized in the trajectory optimization algorithm, trajectory optimization becomes a minimization process. The cost function describes the intention of the trajectory optimization issue and quality of the required motion mostly depends on it. This is the main stage of establishing the

trajectory optimization algorithm because it is clearly related to the issues of accuracy of the trajectory optimization system. Various dynamic cost functions are taken into account in the scientific literature as discussed in Ch. 2. Among the many available cost functions, the most common dynamic objectives to be optimized are [57]:

- Minimum execution time
- Minimum energy (or actuator energy, torque)
- Minimum jerk

The proposed optimization method can take into account any dynamic cost function that can be calculated by using the inverse dynamic model. In advanced implementations, both kinematic and dynamic constraints have to be taken into account to define the cost function of the required system.

The problem being considered here aims to formulate optimum paths which minimize torque and/or energy system inputs. The following expression is used to describe the cost function in this study; G represents the cost (objective) between initial and final postures [9]:

$$G = \int_0^T \left(\sum_{i=1}^{K} g_i^2(t) \right) dt \tag{4.5}$$

where g_i is the required actuator torque to be applied at joint i to achieve the desired motion, and T is the motion duration. Calculation of the cost function in Eq. (4.5) requires the running of the inverse dynamic model for T seconds given the motion by the iterative process of the optimization. Although energy consumption and torque variation were taken into account in this thesis, other quantitative indicators could be considered according to the design objective or it can consist of several sub-cost functions such as time, jerk, or a combination of these variables.

In order to compare the cost result between the theoretical and experimental system, the same trajectories are executed on the Katana 450 robotic manipulator. Katana 450 robotic manipulator provides only the current (s) data for each axis. Assuming that the

torque of a motor is proportional to the current following through it ($g \; \alpha \; s$), the current will be taken into consideration in order to calculate the cost values for experimental studies.

4.1.3 Selection of Path Optimization Technique

This is a crucial topic for the performance of any trajectory planning methodology, and is strongly related to the choice of trajectory and dynamic cost functions. The basic trajectory optimization algorithm should provide a time sequence that can be utilised on any parametrizable trajectory, and it should be applicable in either Cartesian space or joint space coordinates. That is, the trajectory optimization algorithm should be easily utilised on various types of machines such as non-redundant, redundant, hyper-redundant, parallel manipulators, etc., where a reference trajectory is utilised in order to identify the machine's task.

To optimize a given trajectory, ideally computationally efficient methods are preferred, but a large number of parameters and coefficients may adversely affect the results of trajectory optimization, and computational efficiency. In most advanced trajectory optimization algorithms, such as, genetic algorithm, fuzzy logic algorithm, etc., [27], [31], [53] so much time is spent in order to generate and tune the optimization parameters for the algorithm. Although they are difficult to implement and tune, they are capable of producing a global optimum solution at the expense of a larger computational cost.

In this study, it is crucial to have a solver (such as built-in functions) in order to reduce the time expenditure of creating essential optimization algorithm routines. Because of this, the MATLAB Optimization Toolbox function "fmincon", is utilised. The function "fmincon" attempts to determine a constrained minimum of a scalar nonlinear multivariable function starting with initial estimate optimization parameters [159]. This is a sequential quadratic programming based optimizer.

The initial parameters of the optimization algorithm can impress the quality of the trajectory optimization outcome. As is known, "fmincon" is a local optimization approach like the majority of numerical minimization algorithms. Hence, the local

minimum to which the trajectory optimization algorithm converges is dependent on the initial point in the trajectory optimization algorithm. It is obvious that trajectory optimization issues consist of many local minima points. However, minor variations in starting optimization parameters such as shift limits, constraint values, etc., may result in extreme variations in the outcome of cost value. These issues are partly due to flatness of the cost function, but probably more crucially, due to the numerical trajectory optimization procedures utilised to analyse the problems.

Hence, in the current work, feasible initial optimization parameter values will be the input to the proposed optimization algorithm. When this iterative process is terminated, then the final parameters will be used to construct the optimal trajectory.

4.1.4 System Constraints

Constraint trajectory optimization problems aim to accomplish the optimal solution of the cost function under certain constraints, which can be inequality and equality constraints. The constraints can also be classified as follows:

- The system constraints imposed by the robotic manipulator itself due to its kinematic and dynamic limitations.
- The task constraints which are generated by a given task, such as geometric constraints, path velocity and obstacles.

The kinematic and dynamic boundary condition for the trajectory optimization system is shown in Table 4.1. By using these constraints, unrealistic or unreachable motions of the robotic manipulator are automatically avoided in the trajectory optimization procedure. In addition to these constraints, in cases where the B-spline are utilised

Constraints	**Cartesian coordinate**	**Joint coordinate**
Displacement	${x_i}^{min} \leq \mid x_i(t) \mid \leq {x_i}^{max}$	${\theta_i}^{min} \leq \mid \theta_i(t) \mid \leq {\theta_i}^{max}$
Velocity	$\dot{x}_i^{min} \leq \mid \dot{x}_i(t) \mid \leq \dot{x}_i^{max}$	$\dot{\theta}_i^{min} \leq \mid \dot{\theta}_i(t) \mid \leq \dot{\theta}_i^{max}$
Acceleration	$\ddot{x}_i^{min} \leq \mid \ddot{x}_i(t) \mid \leq \ddot{x}_i^{max}$	$\ddot{\theta}_i^{min} \leq \mid \ddot{\theta}_i(t) \mid \leq \ddot{\theta}_i^{max}$
Torque	${g_i}^{min} \leq \mid g_i(t) \mid \leq {g_i}^{max}$	${g_i}^{min} \leq \mid g_i(t) \mid \leq {g_i}^{max}$

Table 4.1: *Maximum and minimum limitations of the constraints in the system.*

to describe the trajectory in Cartesian coordinates, additional non-linear constraints have to be taken into account to ensure that the end point of the manipulator does not fall outside the workspace, i.e., the end-effector motion in Cartesian coordinates has to satisfy the following constraints:

$$\max \sqrt{(f_x^2(t) + f_y^2(t))} \leq \sum_{i=1}^{n} l_i \qquad for \quad 0 \leq t \leq T \tag{4.6}$$

Since f_x and f_y are created from the optimization parameters, the constraints can be checked before calling the inverse dynamics, otherwise, the inverse dynamic solution and hence the trajectory optimization process will fail. If needed, additional constraints can also be considered (such as obstacle avoidance, singularity avoidance etc.) by the trajectory optimization algorithm.

4.1.5 Path Coordinates

In most cases, a robotic manipulator's trajectory is identified in terms of task space coordinates, implementations such as spraying, gluing, arc welding or cutting. The motion of the EEF in task space is clearly predictable and informs the user whether the manipulator's end-effector collides to an obstacle, whereas a joint motion of the manipulator does not ensure such guarantees. On the other hand, there are a number of disadvantages in task space motion. Firstly, the computation of the trajectory is time consuming, that is, the task space data has to be transformed to joint space utilising an inverse kinematics scheme to control the motors. Secondly, it is effortless to visualize the trajectory, but is hard to ensure singularity in the robotic manipulator. Thirdly, a smooth trajectory in task space may require a sudden change in the joint angles during the given motion.

Planning the manipulator trajectory in joint space is done in a similar way to the Cartesian space trajectory planning. However, when a robotic manipulator moves smoothly in joint space does not imply a smooth EEF motion in task space coordinates.

The Cartesian coordinate will be taken into account to show the effectiveness of the proposed alternative cost handling method for the optimum trajectory planning of non-

redundant and redundant/hyper-redundant manipulators in Ch. 5, Ch. 6, respectively. The reason of utilising the Cartesian coordinate for the proposed optimization method is that the implementation of the proposed method in Cartesian space will be more challenging than the joint space due to a number of above mentioned disadvantages.

4.2 Proposed Penalty Algorithm

The cost function calculations involve running the computationally intensive inverse dynamic model, which is time consuming. In conventional methods (such as the *fmincon* function in Matlab) the constraints equations are handled separately as it is seen from Fig. 4-2(1) and the cost function is called regardless of whether the constraints are satisfied or not. In order to improve computational efficiency of the trajectory optimization algorithm, constraints can be handled within the cost function calculations as seen in Fig. 4-2(2). This way, the inverse dynamic analysis is only evaluated when these constraints are satisfied. In order to achieve this, an alternative method of handling constraints within the cost function is introduced in this section [9]. The proposed approach involves running the optimization without the constraints, and checking the constraints within the cost function before calling the inverse dynamic simulation as shown in Fig. 4-7. The process can be summarized as follows [9]:

1. A global variable p is created (the initial values of p is set to zero) to count the number of cost function calls where the parameters do not satisfy the constraint equations [9].

2. During the cost function call, if the constraints are satisfied, the cost value is calculated by calling the inverse dynamic program in accordance with Eq. (4.5) [9].

3. If any of the constraints are not satisfied, an alternative cost value is returned without running the inverse dynamic model as follows [9]:

$$p = p + 1 \quad \text{and} \quad G = bb(1 + p/10) \tag{4.7}$$

where bb is a large base value, and is set to $bb = 10^5$. Eq. 4.7 will ensure that

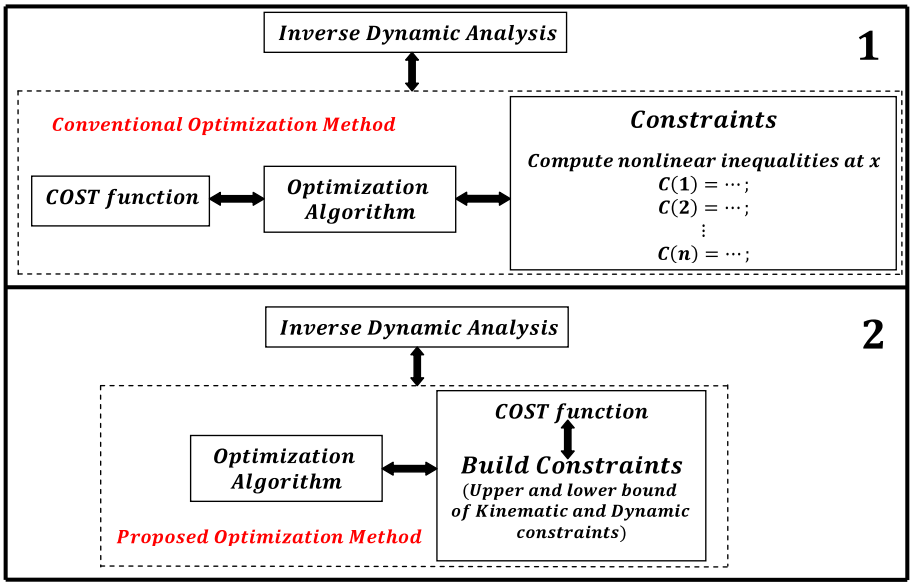

Figure 4-2: *(1) Conventional optimization method; constraints are handled as nonlinear inequalities in the constraint function, (2) Proposed optimization method; constraints are handled in the cost function.*

the alternative cost value will always update and increase at each violation of the constraints to avoid a local minimum to be found outside of the constraints. The procedure of the penalty algorithm has been demonstrated in Fig 4-7.

The automatic update of alternative cost value for each cost function call outside the constraints (independent of the severity of violation) does not distort the original cost function of the trajectory optimization algorithm, instead it has an effect of avoiding some constraints to be treated as more important than others in the optimization algorithm. The value of bb is kept constant during the trajectory optimization process. In theory, this value can be fixed to any arbitrary value, however, in practice, the performance of an optimization algorithm will be depended on the selection of the value of bb. The value of bb should be higher than the worse case cost function value when constraints are satisfied. A value of bb set to $bb = 10^5$ (after some trial and experiments) is selected. In addition to constraints that may be imposed on the position, velocity and acceleration profiles of each joint, it is crucial to ensure that the parametric trajectory functions generated by the trajectory optimization algorithm gives a

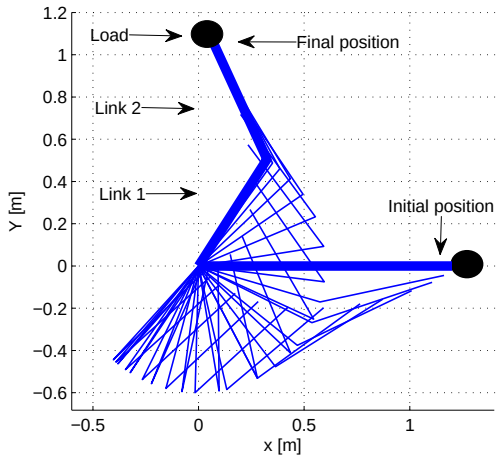

Figure 4-3: *Temporal position of simple two DOF planar robotic manipulator. The robot is constrained to a 2 dimensional environment in x axis and y axis.*

realizable motion within the workspace of the robotic manipulator. Otherwise, the inverse dynamic simulation will fail to run during the cost function calculation and hence the optimization will fail.

The development work for the alternative cost function was carried out utilising the model of the Katana 450 robotic manipulator in Ch. 5 and Ch. 6. However, for clarity of explanation, the proposed alternative optimization method will first be carried out on a simple 2-DOF planar robotic manipulator with revolute joints as shown in Fig. 4-3. The simulation is performed by the program Dysim, which uses the Lagrangian formulation of the equations of motion and is convenient for multi-physics systems. A fifth order B-spline function was utilised to describe the required trajectory. In the simulation, two motors control the motion. Mass centre of gravity of the links are in the middle of the each link and the load mass was selected as 0.1 kg at the end of the second link. The robotic manipulator has two identical links, link 1 and link 2 and physical parameters of the robotic manipulator is given in Tab. 4.2. The motion duration is specified as T=2 s. The manipulator task consists of transporting a load mass from an initial point ($\theta_1 = 0, \quad \theta_2 = 0$) radians and final position ($\theta_1 = 1, \quad \theta_2 = 1$) radians in joint space coordinates. The initial and final velocities and accelerations are zero for all joints. The limits for each actuator in terms of maximum and minimum joint position, velocity, acceleration and torque profiles are given in Tab. 4.3.

Joints	Length	Mass	Inertia	Friction coefficient	Gear ratio
Joint 1	0.6 (m)	1 (kg)	0.01 (kgm^2)	0.4 (Nms/rad)	100
Joint 2	0.6 (m)	$1(kg)$	0.01 (kgm^2)	0.4 (Nms/rad)	80

Table 4.2: *2 DOF robotic manipulator data.*

Contraints	Joint 1	Joint 2
Position (rad)	$-/+3\pi/2$	$-/+3\pi/2$
Velocity (rad/s)	$-/+6$	$-/+6$
Acceleration (rad/s^2)	$-/+25$	$-/+25$
Torque (Nm)	$-/+30$	$-/+30$

Table 4.3: *Limit performance of the 2-DOF robotic manipulator.*

Although the robotic manipulator has two degrees of freedom, the Dysim program selects 8 generalised coordinates (three for each link and two for the load) for the robotic manipulator as follows:

$$q = [x_1, y_1, \theta_1, x_2, y_2, \theta_2, x_L, y_L] \tag{4.8}$$

where x_i, y_i and θ_i are the Cartesian coordinates of the centre of gravity, and the joint angle, respectively, for link i. The system consists of 14 constraints. The Lagrangian function and the dynamic equations of motion including constraint equations and differential-algebraic equations are automatically generated by the program of DYSIM. The program also computes the initial conditions of the dependent coordinates based on the user defined initial position conditions of the user selected two independent coordinates, angle of θ_1 and angle of θ_2. In this case, the angle of links θ_1 and θ_2 was selected as the motion defining variable [9]. The corresponding prescribed non-optimized manipulative task is demonstrated in temporal trajectory position in Fig. 4-4(a), and the optimum trajectory is traced in Fig. 4-4(b). The initial trajectory was a straight line (minimum distance in joint angle) in the joint angle between initial and final position.

The corresponding theoretical cost values for the proposed alternative optimization and conventional optimization algorithm are demonstrated in Tab. 4.4. The cost

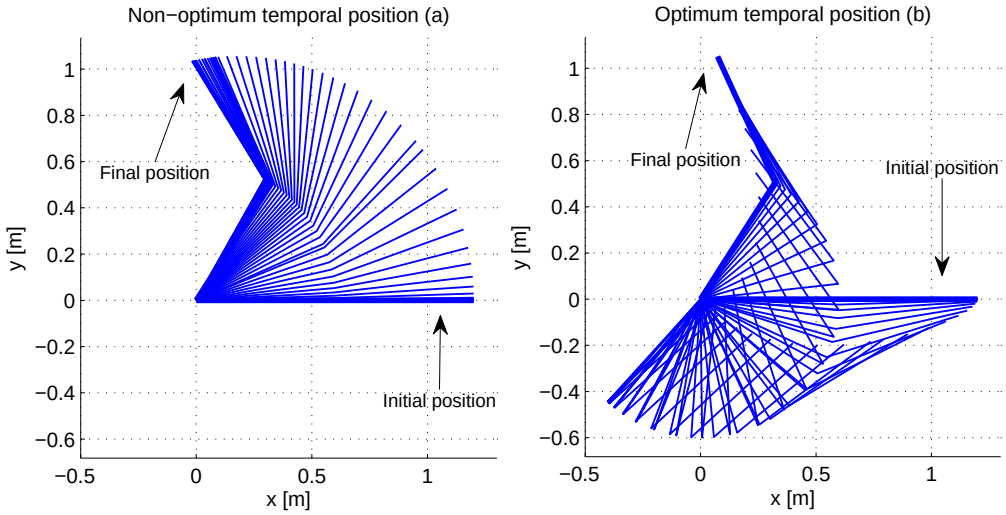

Figure 4-4: *Temporal position of optimized and non-optimized trajectory of simple two DOF planar robotic manipulator.*

function containing a term proportional to the integral of the squared actuator torques and/or energy along the trajectory is considered for the theoretical studies. As it is seen from the cost table, both optimization methods provide the same cost results and maximum energy reduction was observed approximately 31% in both. The corresponding Figure 4-5 demonstrates the computational performance of the proposed algorithm. Because the constraints are handled within the cost function and the constraint violations are punished heavily, the optimization quickly moves away from the parameter space outside the constraints as shown in Fig. 4-5(a) [9]. The computational cost is reduced due to not running the inverse dynamic model in the alternative cost function calculations when the constraints are not satisfied. The high return cost function values in Fig. 4-5(b) correspond to cases where the alternative cost function is utilised and is

Cost	Proposed method	Conventional method
	N^2m^2s	N^2m^2s
Non-optimum Cost (2sec)	888	888
Optimum Cost (2sec)	622	623

Table 4.4: *Cost values for the proposed optimization method and conventional optimization method of the theoretical output of simple two DOF planar robot on joint motion. Theoretical cost calculated from the required actuator torque to be applied at joint i.*

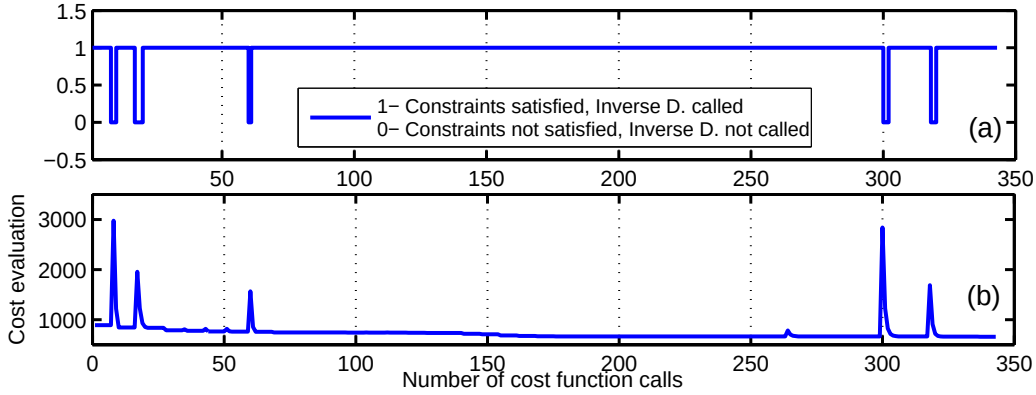

Figure 4-5: *Computational performance of the proposed algorithm, (a) Inverse dynamics calls, (b) Cost function evaluations.*

consistent with Fig. 4-5(a). To further demonstrate this, the advantages of the proposed algorithm over the conventional method of handling the constraints through defining them as nonlinear constraints are demonstrated by the results in Fig. 4-6 [9]. The

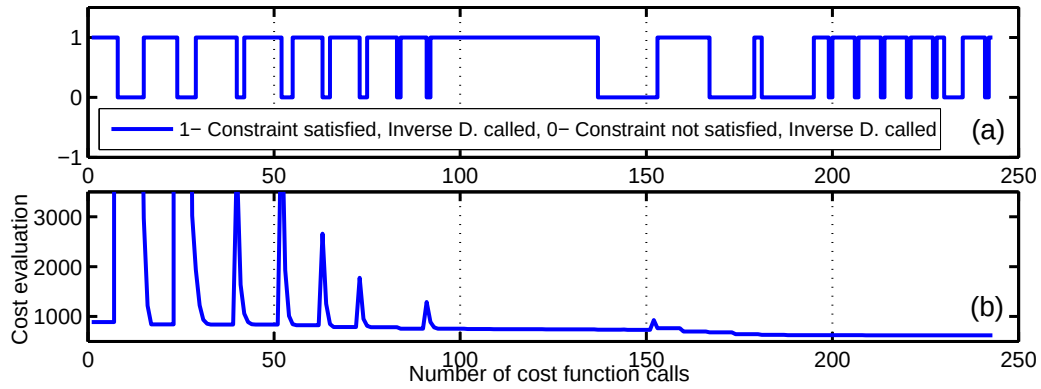

Figure 4-6: *Computational performance of the conventional constraint handling, (a) Inverse dynamic calls, (b) Cost function evaluations.*

same optimization was run with Matlab "$fmincon$" function with the cost function as in Eq. 4.5 and the constraints were specified as nonlinear inequality constraints. As it is shown in Fig. 4-6(a), the optimization algorithm still calls the cost function even when the parameters do not satisfy the constraints in the optimization system. The number of cost function calls with parameter values outside the permissible workspace is significantly increased compared with the proposed method. Furthermore, each cost function call (whether within the workspace or not) requires the inverse dynamic simulations, which has a significant effect on the computational efficiency of the optimization [9].

Also, although the optimization results are very close as shown in Tab. 4.4, the conventional method takes 246 function calls compared with the proposed method with 343 function calls. However, the conventional method calls inverse dynamic twice at each iteration whereas the proposed method calls one or more. It was observed that the optimization algorithm called the cost function even when the parameters did not satisfy the constraints. The number of cost function calls with parameter values outside the permissible workspace was significant (84 out of 246 iterations), resulting in unnecessary solving of the inverse dynamics.

In addition to computational efficiency, in cases where the fifth order uniform B-splines are used to describe the trajectory in Cartesian coordinates, an additional nonlinear constraint has to be added to make sure that the end point does not fall outside the circle of radius (total length of the link 1 and link 2) during the motion. The conventional constraint handling would still call the inverse dynamics model when these constraints were not satisfied. This would cause the inverse dynamics simulation to crash or terminate prematurely as the required motion cannot be physically achieved [9]. The proposed algorithm avoids this problem. Furthermore, the inverse dynamics model used here includes all Cartesian and joint variables as generalized coordinates, hence avoiding the need to perform inverse kinematic calculations after running the inverse model.

4.2.1 Proposed Trajectory Optimization Procedure

After defining the free parameters to be optimized, a trajectory optimization algorithm will be run by an inverse dynamic program based on iterations algorithm to solve the minimization problem. The solution of the optimization problem is obtained using sequential quadratic programming techniques (such as the "fmincon" function in MatLab [159]). The steps of our energy minimization algorithm based on inverse dynamic is shown in Fig. 4-7 and summarized as follows [9]:

1. Before executing the optimization algorithm, all kinematic and dynamic constraints of the mechanism have to be identified. In this example, these constraints

are based on maximum and minimum values of the position, velocity, acceleration and torque values as in Tab. 4.1.

2. Hereafter, the optimization algorithm will begin from suitable and feasible initial con- ditions (straight-line for non-redundant case and randomly for redundant/hyper-redundant case) for a given initial and final position in the desired coordinate system and the duration of motion [9].

3. Before the cost function calculation algorithm runs the inverse dynamic program, the optimization algorithm will check the kinematic constraints boundary conditions (such as position, velocity and acceleration profile as in Tab. 4.1) of the system. In this step of the algorithm, two situations can occur [9].

   (a) If the kinematic constraints are not satisfied and have violated the boundary conditions, the inverse dynamic analysis is not run. Kinematic and dynamic constraint equations of the robot have been included in the cost function. Therefore, these kinematic boundary conditions in the cost function will be punished heavily by utilising an alternative cost function (as in Eq. 4.7) without taking into account in the optimization algorithm. After all this, the optimization algorithm will go back to step 2 in order to find minimum cost value in the system [9].

   (b) In other cases, when the limitations of the kinematic constraints are satisfied, inverse dynamic will then be run in order to ensure the torque limitation between the maximum and minimum torque values. In this case, two situations can occur: if the torque limitation is not satisfied with the condition of maximum and minimum torque values, the dynamic constraint equation will also be punished by alternative cost function without taking into account the result of the cost value in the optimization algorithm. Hence, the optimization algorithm will continue with step 2. In other cases, if the torque limitation is satisfied, cost value of the system will be calculated by the inverse DYSIM dynamic program. The output of the inverse dynamic will be the new cost value of the system [9].

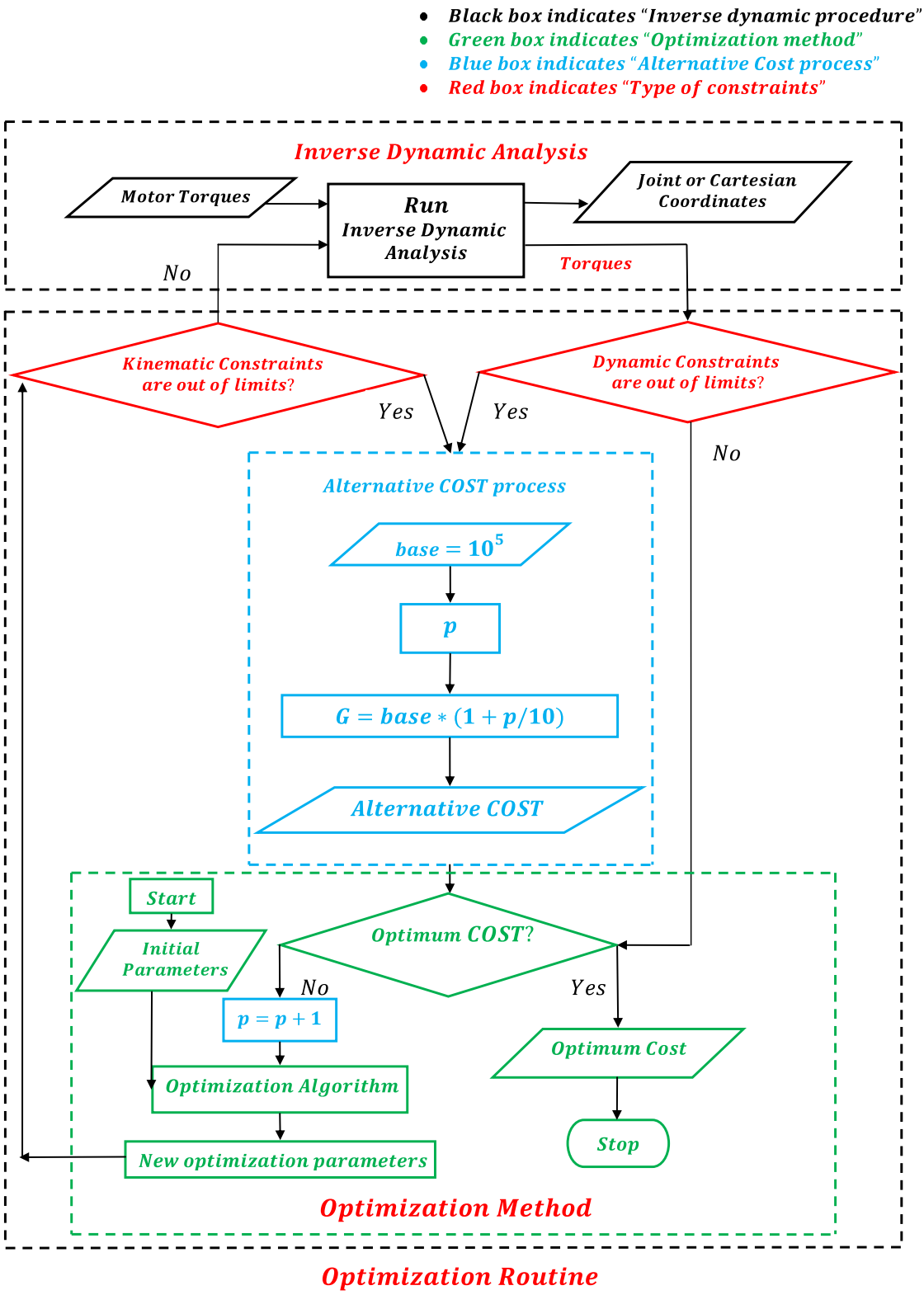

Figure 4-7: *Optimization procedure with the proposed penalty algorithm.*

4. In step 3, if the cost value is optimum, the optimization algorithm will be terminated and this cost value will be the optimum cost in the desired system. If it is not, parameters of the cost value will then be the inputted to the optimization algorithm and the optimization procedure will continue with the step 2 [9].

5. This procedure will continue until the optimization algorithm finds the lowest cost value. The procedure of the optimization algorithm and also the proposed penalty algorithm are shown in Fig. 4-7 [9].

4.3 Concluding Remarks

This chapter has dealt with the procedure behind the trajectory optimization method utilising a proposed alternative cost handling methodology, which is applicable for various types of robotic manipulators. In order to define the desired trajectory, a fifth order uniform B-spline function was constructed and hence the continuity of velocity and accelerations were guaranteed. An inverse dynamic model of a two degree of freedom manipulator was performed in order to demonstrate the effectiveness of the proposed method by using Lagrangian dynamics and an in-house software package DYSIM. In the proposed alternative cost handling method, all of the constraints are built into the cost function. Therefore, computational complexity is reduced by preventing the running of inverse dynamic analysis when all constraints are not satisfied.

In addition to this, crucial stages of trajectory optimization procedure and minimum energy consumption trajectories were also discussed, such as identifying the parametric path function, selection of a cost function, defining the optimization technique, kinematic and dynamic system constraints and selection of path coordinates. To derive the desired trajectory, a multi-parametric trajectory optimization method was utilised. In the optimization algorithm, a nonlinear constrained optimization algorithm is used as the optimization method. Minimum energy consumption trajectories are calculated by considering the main constraints imposed on the robotic kinematic and dynamic performance. Actuator torques and current have been taken into account for the formulation of the cost function for the simulation study and experimental work, respectively.

CHAPTER 5

EXPERIMENTAL RESULTS FOR NON-REDUNDANT ROBOT MANIPULATORS

In Ch. 4, the procedure behind the optimal trajectory planning method for various types of robotic manipulator was presented. The proposed alternative optimization method was also investigated and performed in order to handle the kinematic and dynamic constraints equations effectively during the required task.

In this chapter, the utilisation of the proposed alternative optimization method and also trajectory optimization procedure will be demonstrated based on inverse dynamic analysis. A number of numerical simulations followed by the experimental outcomes, have been carried out to verify the validity of the proposed alternative optimization approach. To demonstrate how the proposed optimization method works, a model of a Katana 450 two-link robotic manipulator system has been chosen for the non-redundant system study as shown in Fig.5-1, because of its simplicity and ease of demonstration. Despite its simplicity, the robotic manipulator lets us demonstrate the crucial aspects of the optimal trajectory planning algorithm. However, the proposed optimization algorithm and also the trajectory optimization methodology are similarly applicable to more complicated robotic manipulators including a larger number of links.

A theoretical and also experimental study of the Katana 450 is carried out by the program of Dysim. To run a two link non-redundant manipulator system, link 2 and

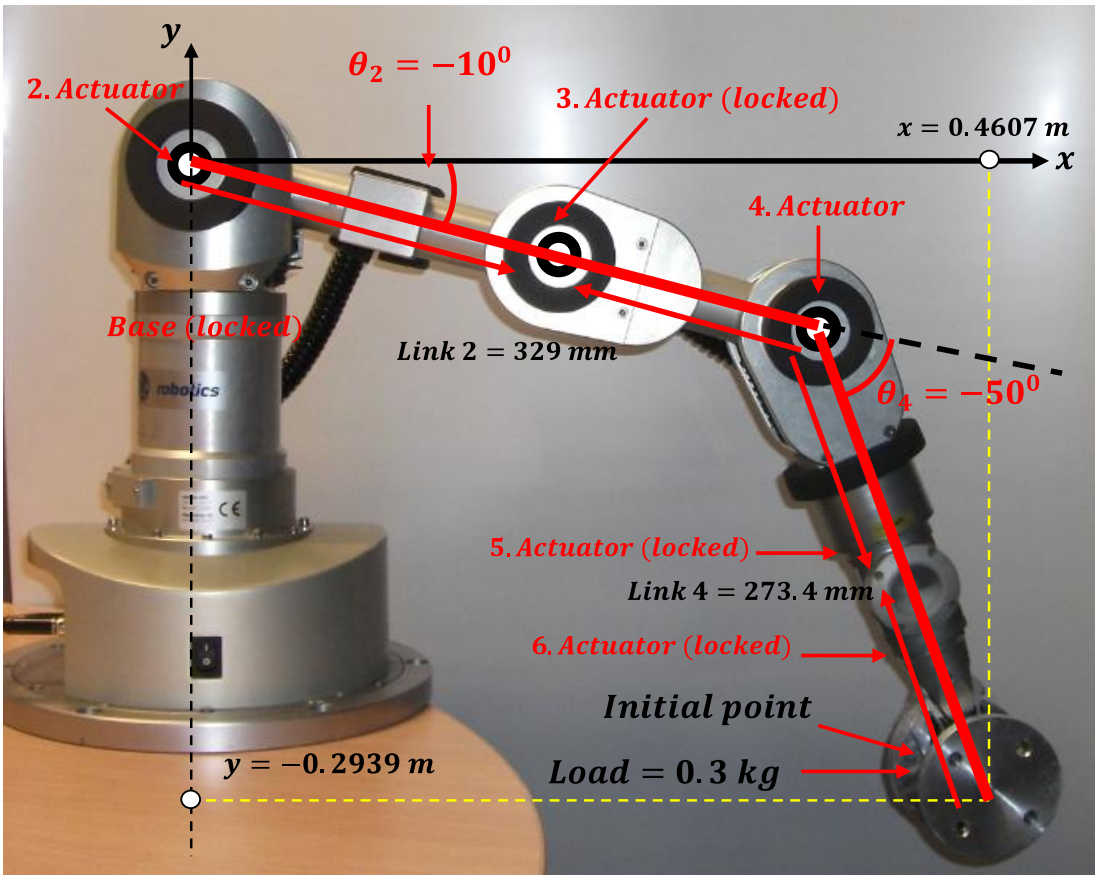

Figure 5-1: *Katana 450 robotic manipulator (rotary joints 2 and 4) based on link 2 and link 4.*

4 of the Katana 450 robotic manipulator (rotary joints 2 and 4) were taken into account and modelled in the previous Chapter 3. In order to create a dynamic model, all of the geometrical and inertial link parameters are listed in Tab. 3.1. The motor specifications and also angle limitation of the Katana 450 robotic manipulator are given in Tab. 3.3 and Tab. 3.2, respectively. In this chapter, the proposed alternative optimization method as well as the off-line trajectory planning algorithm were performed successfully. The success of handling the constraints effectively in the proposed alternative optimization method was demonstrated clearly in various numerical and also experimental studies. In order verify the validity of the proposed optimization method as well as the characteristics and limitations of the modelled Katana robotic manipulators, three exemplar types of trajectory motion with different settings (various durations of motion) are introduced and compared successfully in various theoretical

and experimental simulations, namely:

- Straight-line on end-effector (EEF) motion
- Straight-line on Joint motion
- Optimal motion

All of the theoretical simulations are tested experimentally on the Katana 450 manipulator. To make a comparison between the varied types of manipulator trajectories, same initial and final positions were selected, and the duration of motion was increased from 2 seconds to 8 seconds by a factor of 2.

All of the experimental data was provided from the Katana 450 manipulator's axes. The actual manipulator data will be compared with the demand data, but the actual data (such as encoder position, encoder velocity and encoder time) is not identical with the data of demand motion obtained by the reference manipulator trajectory. Therefore, the experimental outcomes of the system were all derived from raw experimental data recorded from the Axis Scope software of the Katana manipulator during the required motion. Axis Scope software of the Katana robotic manipulator only ensures the position, velocity and current data in order to analyse and compare the demand and actual trajectories. As previously mentioned in Ch. 3, acceleration and jerk profiles are not implemented for the experimental results of the system.

The proposed optimization method was tested by dynamically simulating a Katana 450 robotic manipulator. In order to ensure that the simulation environment is as precise as possible, the same set of trajectory movements are executed on the robotic manipulator as in simulation and compared the total energy outputs by means of Eq. 4.5. The trajectories are identified by fifth-order B-spline functions, as in section 4.1.1, and transformed to the cubic polynomials, as in section 3.3.1, in order to put these trajectories into practice by utilising the Katana 450 robotic manipulator. The coefficients of the cubic polynomials can then be utilised to input into Katana's axis controller for robotic motion. In order to compute the total energy consumed for the robotic manipulator, the manipulator trajectories were executed 5 times, and the

current required in each of the manipulator joints recorded and averaged; this ensures a value approximately proportional to the torque profile. This mean current is then utilised as torque (g_i) in Eq. 4.5, in order to compute the total energy consumed by the robotic actuators throughout the required movement. In addition to this, the error analysis of the generated manipulator's trajectories was also investigated to determine the quality of the trajectories in section 5.4. The quality of the trajectory depends on the quality of the tracking error for the desired system.

From the next section, the exemplar types of trajectory motions with different settings (various durations of motion) will be performed individually and in detail by means of theoretical and also experimental studies in the next three sections.

5.1 Straight-line on Cartesian Coordinates

A straight line path on end-effector (EEF) motion (with constant orientation) with various durations of motion were taken into account the in Cartesian coordinate system. The robotic manipulative task consists of transporting a load mass from an initial position to a final one as demonstrated in temporal motion position in Fig. 5-3. The initial and final velocities are zero for all manipulator's links. The robotic manipulator is asked to carry the load between the initial point ($x = 0.4607, \quad y = -0.2939$) m and final point ($x = 0.4607, \quad y = 0.2939$) m.

The straight line trajectory of the robotic manipulator is identified by a parametric equation which utilizes a fifth order B-spline function to represent the Cartesian coordinates of the end-effector's path along the trajectory. The end-effector of the robotic manipulator contains two possible joint configurations (based on link 2 and 4 of the Katana manipulator) in the joint space as shown in Fig. 5-2. The path 2 was selected for the theoretical and experimental studies on straight-line motion.

In this motion, the initial point of the end-effectors is selected as almost the maximum capable limit of the link 2 and starting from 10° degree as is seen from the Fig. 5-2. In addition to this, the final position of the end-effectors was chosen as its maximum achievable straight-line distance from the initial point for this robotic ma-

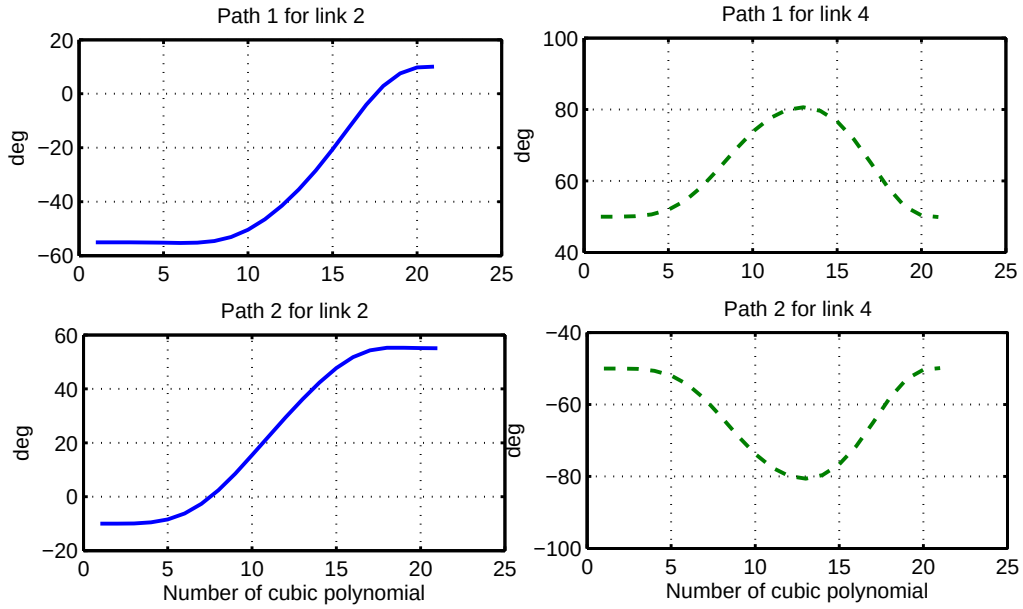

Figure 5-2: *Two possible solutions for two link non-redundant planar robotic manipulator for straight-line motion.*

nipulator configuration as shown in Fig. 5-3. Transfer between initial and final position is executed in a smooth way and without violating imposed kinematic and dynamic constraint conditions.

Figure 5-4 demonstrates the comparison between the theoretically simulated velocity profiles and the experimentally recorded output velocity profiles for the desired motion based on links 2 and 4. The maximum velocity limits for joint 2 and 4 are 73 deg/s and 134 deg/s, respectively and are demonstrated in Figure 5-4.

Inherently, the results obtained from the theoretical implementations may not be completely identical with the results which are obtained from the experimental studies due to the uncertainty involved in the parameters of the actual manipulator system and modelling the robotic manipulator. Therefore, one would expect that the experimental implementations would provide outcomes slightly different to the theoretical outcomes.

As previously mentioned in the maximum velocity check experiment in section 3.3.3 of the Ch. 3, the Katana robotic manipulator reaches the manufacturer's maximum allowable velocity limit profile at 1.8 sec motion with 0.3 kg load and eventually violates its actuator limits. Therefore, two second motion is selected as the fastest achievable

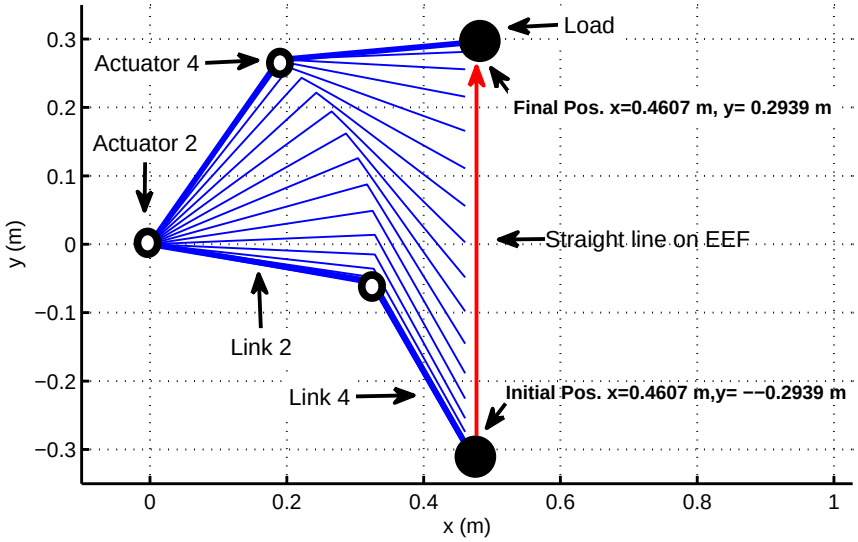

Figure 5-3: *Temporal position of straight line on EEF motion of two link non-redundant robotic manipulator. The robot is constrained to a 2 dimensional environment in x axis and y axis.*

motion for the theoretical and experimental studies of a non-redundant manipulator.

As is seen from the 2 seconds manipulator motion of Fig. 5-4, the maximum velocity profile achieved by the reference velocity (theoretical) profile of link 2 of cubic trajectory is clearly higher than that achieved by the corresponding experimental output velocity profile of link 2. Although the theoretical velocity profile of link 2 for two seconds motion can be achievable successfully without violating the actuator limit in the theoretical simulation, due to the reasons mentioned above such as an uncertainty and modelling issue, the velocity saturation was also determined to occur in the 2 second motion of an experimental study in joint 2 with the 0.3 kg load. The effect of the load would cause the velocity profile of the joints to deviate from its demand velocity profile. In this case, the output velocity profile of link 2 in two second motion is acting as violating its actuator velocity limit for this motion.

As it is seen from the figure, the velocity profile of 2 second motion was determined to deviate the velocity furthest from the demand, followed by the slower motion profiles of 4, 6 and 8 seconds. Interestingly, after velocity deviation of output velocity of link 2, the final path and velocity of the Katana robotic manipulator were on the correct

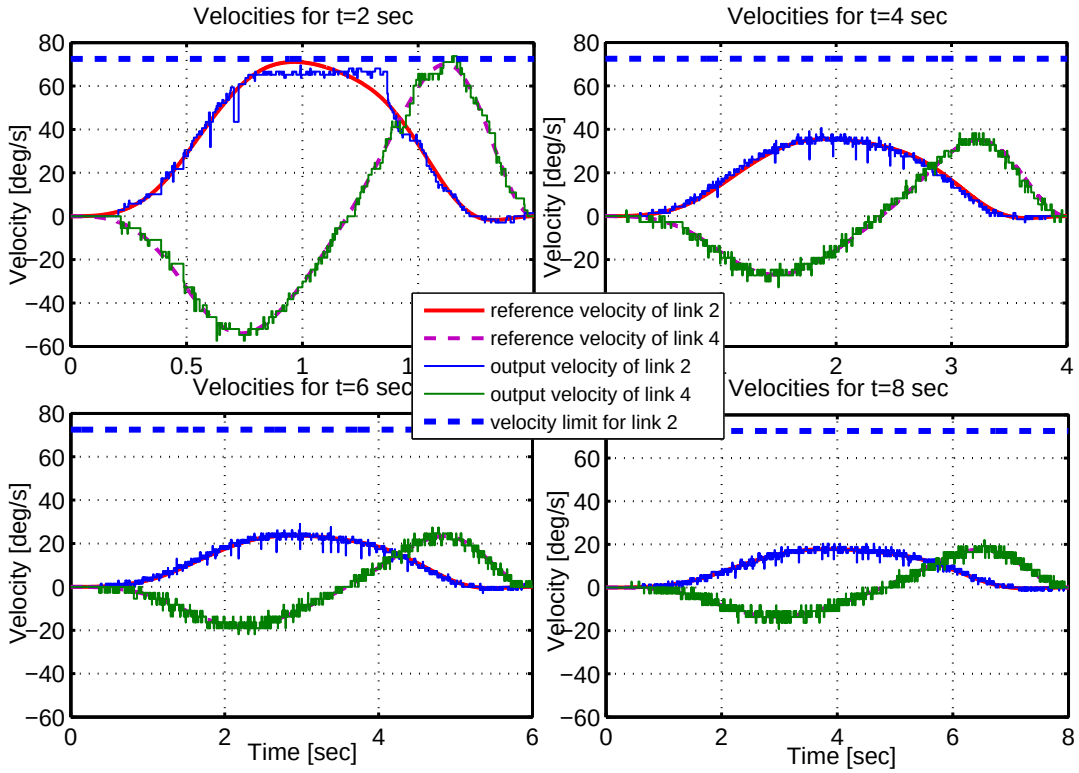

Figure 5-4: *Comparison between demand (theoretical) velocities and actual recorded output velocities (experimental) with different duration of motion for a straight-line motion of the EEF.*

position. The reason for this is that the trajectory of the robotic manipulator was described by the cubic splines. The controller of the Katana robotic manipulator is tracking each section of the cubic polynomials was determined to be able to compensate the velocity shortfall profile during the saturation in the next section of the cubic polynomials. This is demonstrated by the experimental output velocity of link 2 for 2 second motion in Fig. 5-4. The controller of the Katana robotic manipulator attempts to preserve the maximum velocities after velocity saturation so as to compensate for its positional error. This experimental output demonstrates that the feedback of the control system of the Katana robotic manipulator was handling the control situation properly in order to reduce the tracking error in the required system. For the 4, 6 and 8 seconds of durations of velocity profiles, the experimental results support the theoretical simulation results, and identical velocity profiles between the reference and actual velocities have been observed clearly.

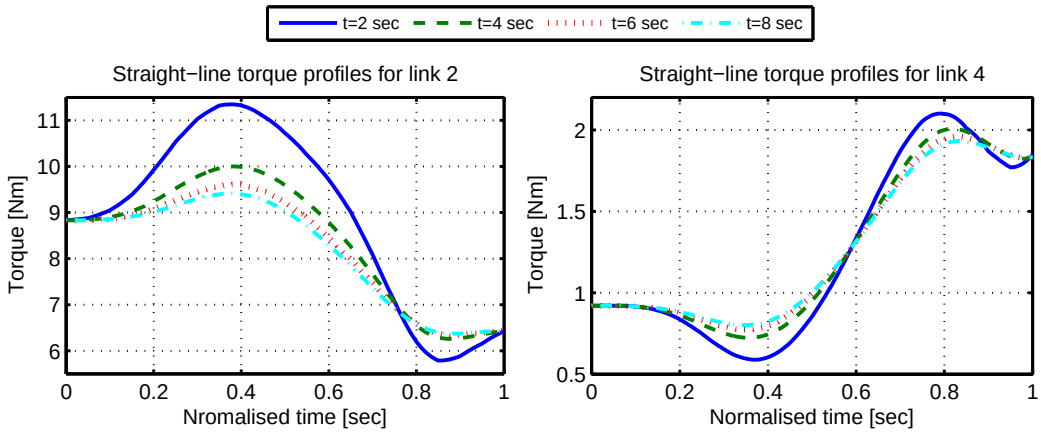

Figure 5-5: *Simulated planned trajectory of straight-line on EEF motion for various duration of motion with torque profiles (normalised time for direct comparisons between the torque profiles of different type of trajectories).*

Furthermore, the velocity profile in Fig. 5-4 indicates that all the velocity profiles demonstrate a clear correlation between the output torque profiles. While executing the desired task in straight-line on EEF motion, essential torque requirements are given in Fig. 5-5. The non-optimized torque profile of link 2 has large and similar torque magnitudes with various durations of motion. In this motion, in order to keep the manipulator's end-effector position on the straight-line trajectory, link 4 of the Katana manipulator moves toward to the gravity direction for a short period of time, while the robotic manipulator lifts the link 2. Therefore, the torque profile of link 4 has a short fall in its magnitude due to the motion profile of link 4. As is seen clearly from the Fig. 5-5, changing the durations of motion of the system results in reducing the torque magnitude of the link 2 during the required motion.

The recorded non-optimum current profiles of the various durations of motion of the experimental results are also demonstrated in Fig. 5-6. In order to compute the current of each manipulator's actuator for each duration of motion, the experimental required motions were executed 5 times, and the current required in each of the actuators was recorded and averaged from 5 for each sampling of the simulation time. As is seen from the figure, all of the current profiles have approximately similar behaviour in the current requirements of each actuator for the various durations of motion. The

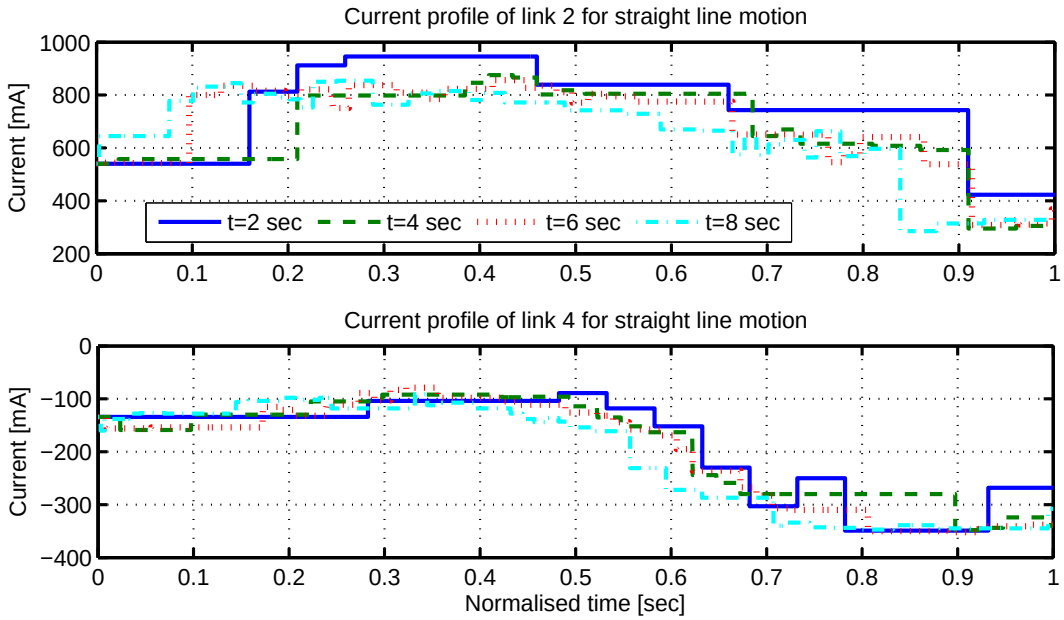

Figure 5-6: *Experimental comparison of current profiles of straight-line motion with duration of motions.*

current profile of link 2 for the two second motion is consistent with the torque profile of link 2 in the same duration of motion. For this duration of motion, the current for link 2 has a bigger torque magnitude than the other and is increased until 0.5 normalised second and measured approximately between 500 mA and 930 mA during that motion. As expected, the current requirements for the other durations of motion profiles are decreasing inherently during the required motion. This comparison clearly demonstrates that the experimental results strongly support the theoretical torque outcomes.

Time	Straight-line (Simulation)	Straight-line (Katana)
	N^2m^2s	amp^2s
Cost(2s)	171	105
Cost(4s)	299	135
Cost(6s)	430	173
Cost(8s)	561	212

Table 5.1: *Cost values of the theoretical and experimental output of straight-line on EEF motion with various duration of motion. Theoretical cost calculated from the required actuator torque to be applied at joint i. Experimental cost calculated provided current data for each axis.*

The total energy consumed by the robotic actuators is given by the theoretical and experimental studies in Tab. 5.1. These cost values can be accurately calculated by the cost function which is given by Eq. (4.5). Naturally, the consumed energy in the actuators is increased by increasing the execution time for the required task. This effect is demonstrated clearly as shown in Tab. 5.1. As it is seen from the Tab. 5.1, significant increments on the cost values appear to be due to increasing the duration of the motion from 2 seconds to 8 seconds for the desired trajectory due to carrying the weight.

5.2 Straight-line on Joint Coordinates

A straight line joint motion (with constant orientation) with various motion durations was taken into account in this section. The robotic manipulative task consists of transporting a load mass from an initial position to a final one as shown in the temporal position in Fig. 5-7. In this example of the straight line joint motion, the robotic manipulator is at rest initially and stops at the end of the trajectory, i.e., the initial and final velocities are zero for all manipulator's links. The robot is asked to carry the load between the initial position ($\theta_2 = -10^0, \quad \theta_4 = -50^0$) degree and final position

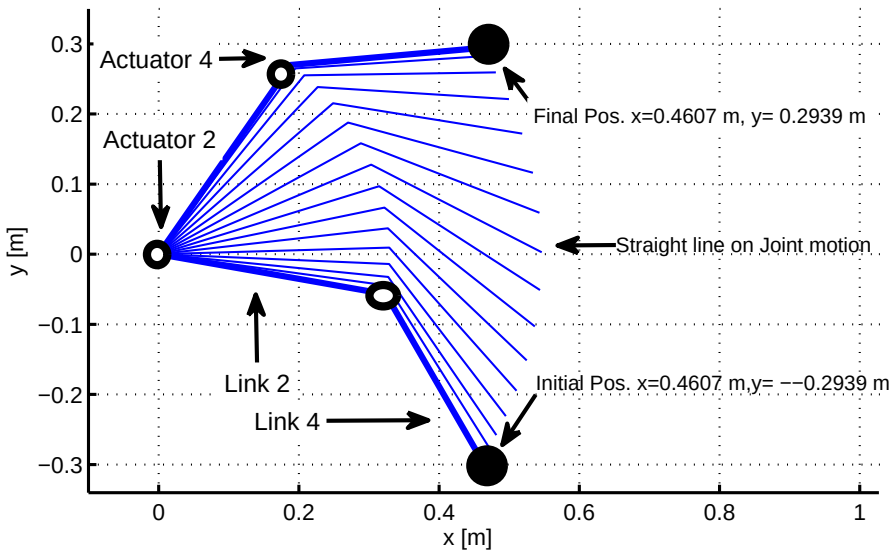

Figure 5-7: *Temporal position of straight-line on joint coordinate of two link non-redundant robotic manipulator.*

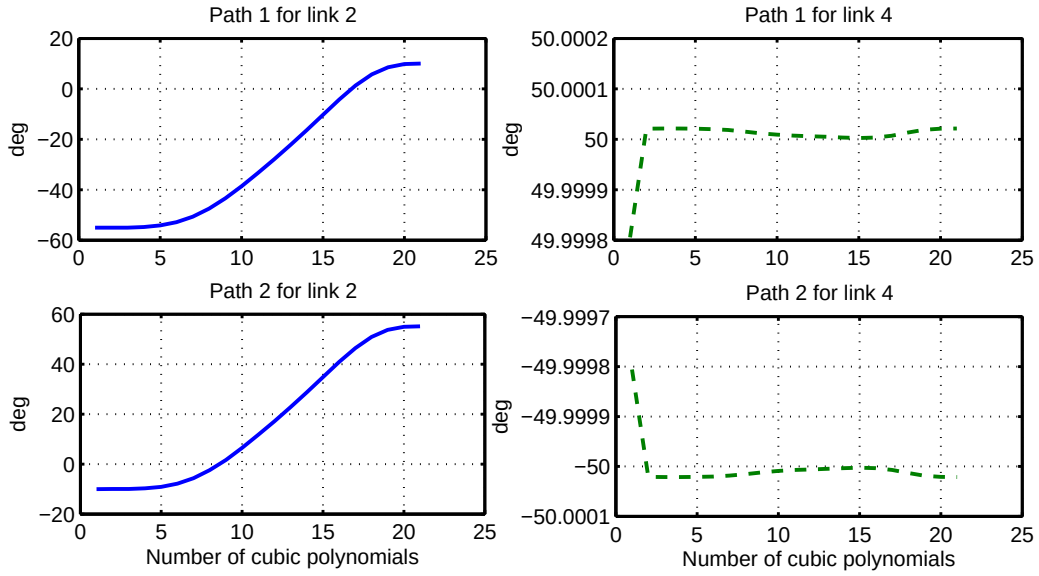

Figure 5-8: *Two solutions for two link non-redundant planar robotic manipulator for straight-line joint motion.*

($\theta_2 = 55^0, \quad \theta_4 = -50^0$) degree as in section 5.1. Fig. 5-8 gives the two possible angle configurations for the desired system. The path 2 was selected for the theoretical and experimental study, the same as in the straight-line EEF motion. For a given initial and final point, link 4 of the robotic manipulator did not change its position while the manipulator was executing its task.

Fig. 5-9 demonstrates the comparison between the simulated velocity of joint motion profiles and the experimentally recorded velocities (Katana output), for joints 2 and 4 respectively for four different motion durations. The demanded velocities of the links do not violate the speed constraints with the velocity of link 4 being zero during the motion. When the velocity comparison has been made between the straight-line on EEF motion in Fig. 5-4 and straight-line on joint motion in Fig. 5-9, velocities of the straight-line on EEF motion are faster than the straight-line on joint motion for the same motion durations. Unlike the maximum velocity profile of joint 2 in a straight line EEF for two second motion as given in Fig. 5-4, the velocity of joint 2 in joint space motion for two second has a smaller speed profile. That is, this result clearly demonstrates that the desired motion in joint space can be achieved faster (without violating the actuator's limit) than the motion in Cartesian space by reducing the

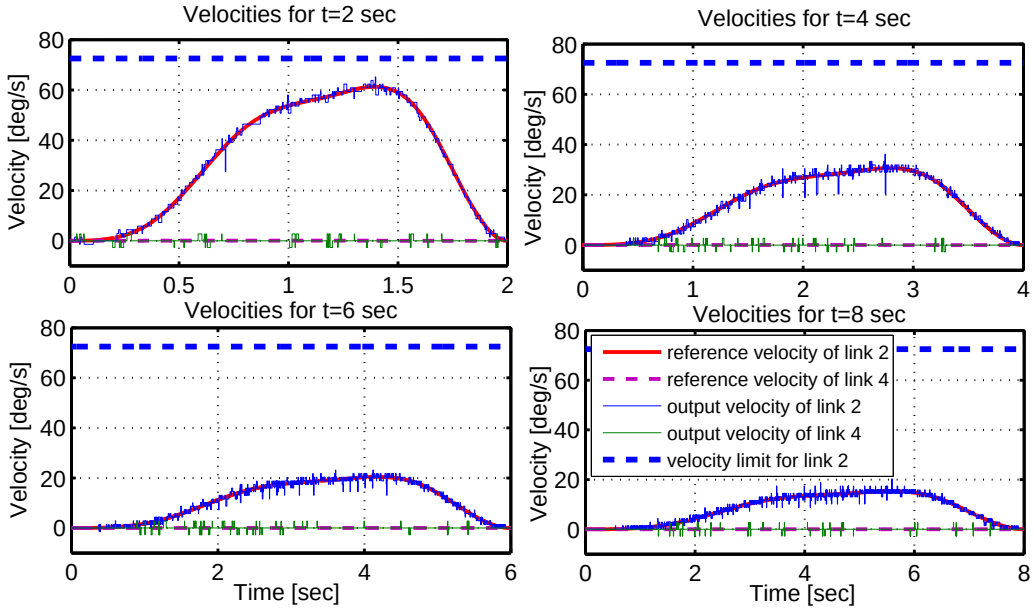

Figure 5-9: *Comparison of speed with different duration of motion (Straight-line on joint motion).*

current motion's duration.

The corresponding torque requirements are shown in Fig. 5-10. In order to complete the task, this motion has required slightly bigger torque profiles during the motion than the torque profiles of straight-line on EEF motion as in Fig. 5-5. Apart from the torque profile of link 2, the other torque profiles of 4, 6 and 8 seconds have almost similar torque behaviour during the required motion. As expected, the torque profiles of link 2 are decreasing due to increasing the durations of motion. Joint motion torque profiles of link 4 support the velocity profiles of link 4 and a smooth increase was observed during the required motion. As is seen from the corresponding current profiles in Fig. 5-11, it is consistent with the velocity and torque profiles in Fig. 5-9 and Fig. 5-10, respectively. The current profile of link 2 for two second motion duration is consistent with the torque profile of link 2 for same motion duration and has a bigger current profile amongst the other. The current profiles of link 4 are also consistent with the torque profile of link 4. Therefore, the simulation results support the experimental outcomes, and there is a strong correlation between them. The corresponding non-optimized cost values of the theoretical and also experimental results are shown in Tab. 5.2. Excessive growth of the cost values of the straight-line joint motion are shown to be due to increasing

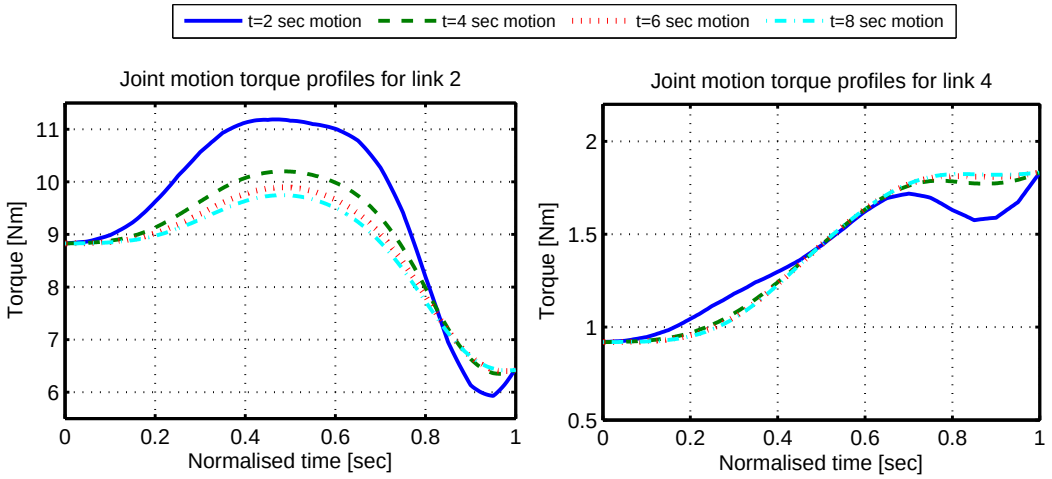

Figure 5-10: *Simulated planned trajectory of straight-line on joint motion for various duration of motion with torque profiles (normalised time for direct comparisons between the torque profiles of different type of trajectories).*

the duration of the motion from 2 seconds to 8 seconds for the desired system. The differences in the cost functions between the straight-line EEF motion and joint motion are due to slightly bigger torque requirements for the straight-line joint motion profile. The other explanation is that the end-effector of the manipulator and also load move further away from the origin during the motion, and hence the torque is higher on the actuator 2 in order to execute the motion.

Cost	Joint motion(Simulation)	Joint motion(Katana)
	N^2m^2s	amp^2s
Cost(2s)	189	126
Cost(4s)	334	152
Cost(6s)	481	189
Cost(8s)	629	234

Table 5.2: *Cost values of the theoretical and experimental output of straight-line on joint motion with various duration of motion. Theoretical cost calculated from the required actuator torque to be applied at joint i. Experimental cost calculated provided current data for each axis.*

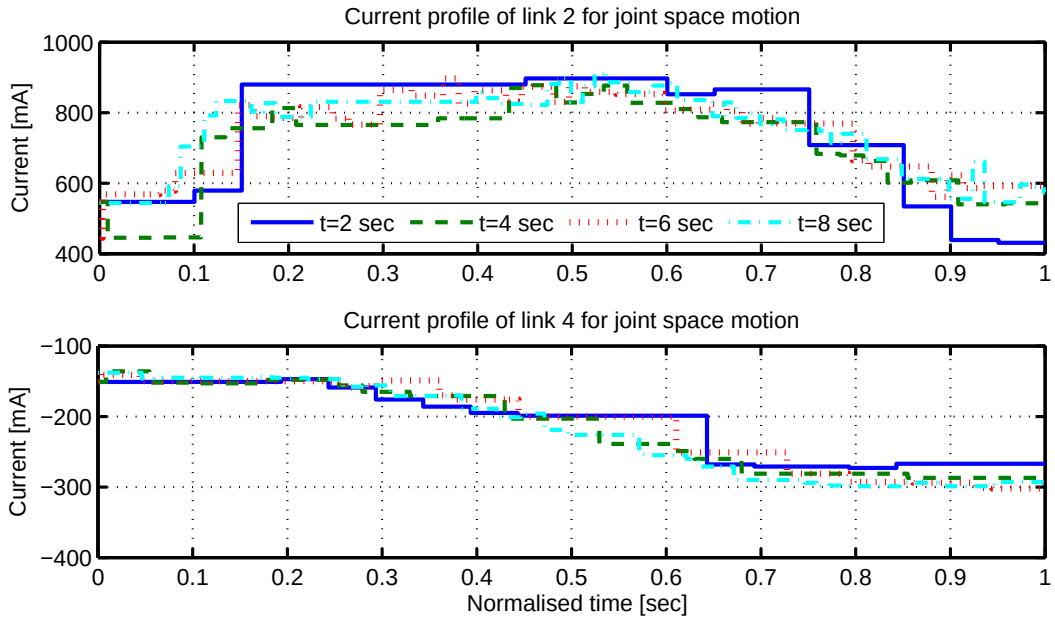

Figure 5-11: *Experimental comparison of current profiles of joint motion with duration of motions.*

5.3 Optimum Trajectory Planning on Cartesian Coordinates

This section deals with the optimal trajectory planning based on minimum torque and/or energy consumption criteria. The proposed optimization method considers an inverse dynamic model of the Katana 450 robot manipulator. This system was modelled by utilising the Lagrange's equation of motion as described in the previous Ch. 3 and Dysim software package is utilized to automatically develop the dynamic conditions of the required system.

The optimal manipulator trajectory planning uses fifth-order uniform (with control points uniformly distributed along time scale) B-spline functions to represent the Cartesian coordinates of the manipulator's end effectors as described in section 4.1.1. The aim of this trajectory optimization is to create the reference inputs for the control system to follow a specified path that requires the least amount of energy amongst several possible paths. In a non-linear constrained trajectory optimization problem, the optimal motion planning is converted into a parametric non-linear constrained tra-

jectory optimization issue by modifying joint variables as a function of set parameters that can then be optimized.

In order to initiate the proposed optimization algorithm, initial values of the free optimization parameters of the uniform fifth order B-spline function have to be specified. The free optimization parameters are selected according to the required motion of the system. As for all iterative routines, the selection of a convenient initial solution is very important as a wrong selection would affect the computation time and convergence of the optimization algorithm. One way to overcome this problem for a non-redundant manipulator is to start with an approximate straight line motion between the initial and final positions.

In this section, optimum proposed trajectory planning with different durations of motion was executed. However, in order to make a comparison the optimum trajectory result with the straight line EEF motion and also straight line joint motion, the tests were done with the same joint configuration as given in section 5.1 and section 5.2. The robotic manipulator task consists of transporting a load mass of 0.3 kg from an initial point ($x = 0.4607, \quad y = -0.2939$) m to a destination ($x = 0.4607, \quad y = 0.2939$) m in Cartesian space by taking into account the kinematic and dynamic constraints imposed on the Katana 450.

The reason of selecting a Cartesian coordinate system for the implementation of optimal trajectory planning is that the handling of the kinematic and dynamic constraints in Cartesian space is more difficult than the handling of them in joint space coordinate. Because of the kinematic and dynamic constraints need to be handled effectively during the required motion for the success of the optimization outcomes. Hence, in order to demonstrate the effectiveness of the proposed alternative optimization method, the Cartesian space is selected for the implementation of optimal trajectory planning for non-redundant robotic manipulator.

In this system, the geometric path, the kinematic and dynamic constraints are the inputs, the trajectory of end-effectors is the output that is expressed as a time sequence of position, velocity and acceleration profiles. The cost function is given by the trajectory motion duration integral of the squared required actuator torques as

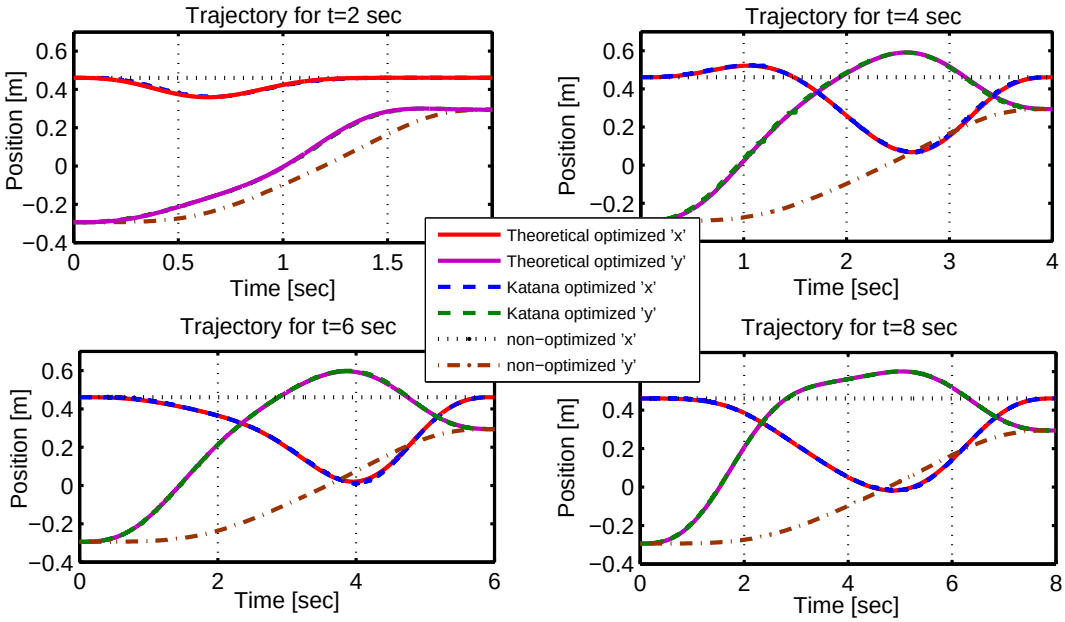

Figure 5-12: *Comparison of the theoretical non-optimum and optimum and optimum experimental positions of the required trajectories.*

given in section 4.1.2 for the simulation study. The trajectory optimization routine is based on inverse dynamic analysis which requires an acceleration profile of the motion as identified by fifth-order uniform B-spline functions, which ensure continuity of velocity and acceleration profiles and provide zero velocity and acceleration for the start and end position of the required motion for the simulation study. In the simulation, two motors control the desired motion. All of the essential physical parameters of the robotic manipulator were identified in a previous Ch. 3, and are utilised to create the inverse dynamic model. The viscous friction effects of the joints are also included with a coefficient of friction of 1.8 Nms/rad and 0.39 Nms/rad for link 2 and link 4, respectively. The duration of motion was changed from 2 sec to 8 sec by a factor of 2. Although the robotic manipulator has two degrees of freedom, the Dysim program selects 8 generalised coordinates (three for each links and two for the load) for the manipulator as follows:

$$q = [x_1, y_1, \theta_1, x_2, y_2, \theta_2, x_L, y_L] \tag{5.1}$$

where x_i, y_i and θ_i are the Cartesian coordinates of centre of gravity, and the joint angle, respectively, for link i. The system consists of 14 constraints. The Lagrangian function

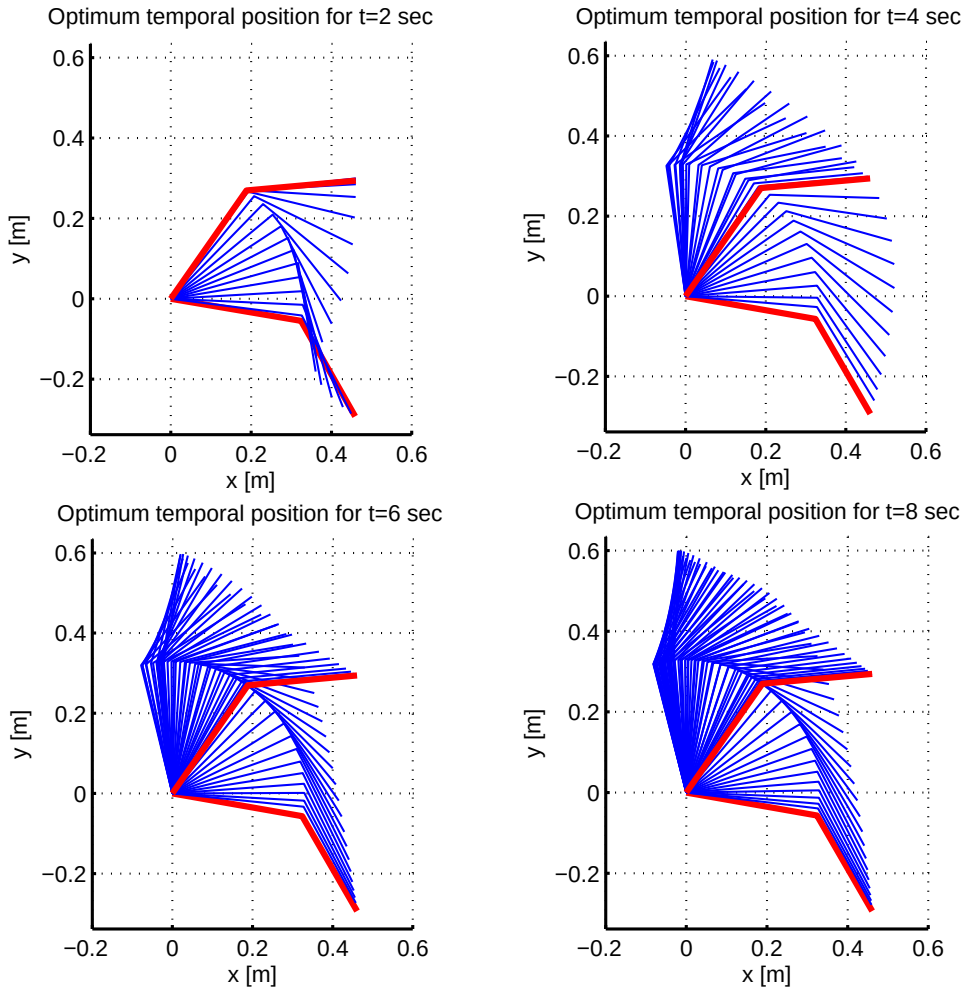

Figure 5-13: *Temporal position of optimized trajectory of two link non-redundant robotic manipulator with different duration of motion.*

and the dynamic equations of motion including constraint equations and differential-algebraic equations are automatically developed by the DYSIM software program. The program also computes the initial conditions of the dependent coordinates based on the user defined initial position conditions of the user selected two independent coordinates, angle of θ_1 and angle of θ_2. In this case, x_L and y_L were selected as the motion defining variables. To initiate the trajectory optimization algorithm, the initial path was selected as a straight-line trajectory in the Cartesian space between the initial and final point of the manipulator trajectory. The prescribed manipulative task and the obtained optimum theoretical and experimental paths are demonstrated in Fig. 5-12.

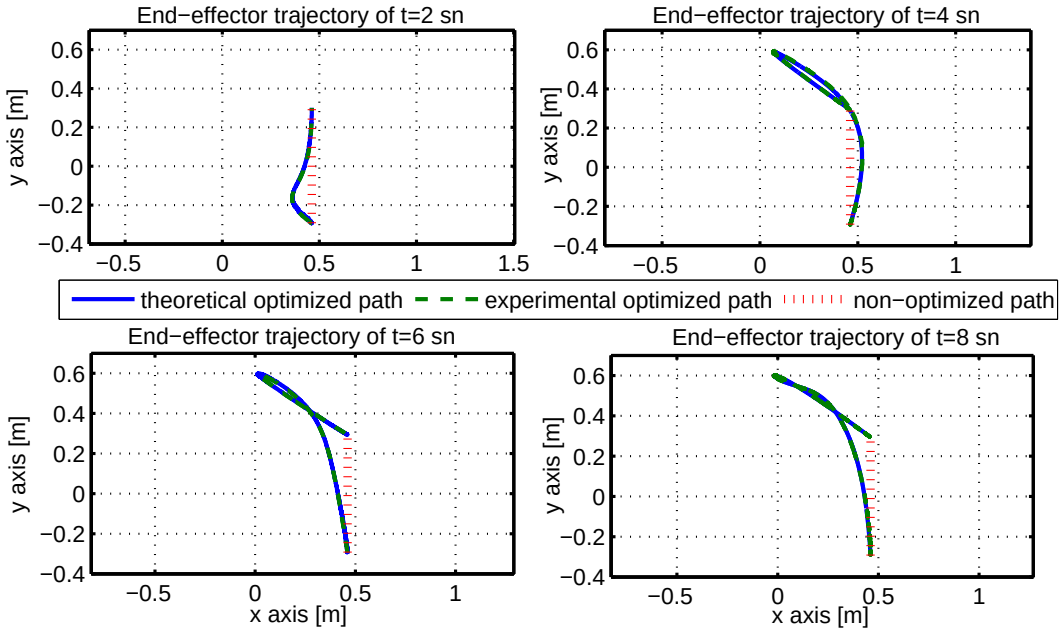

Figure 5-14: *Motion trajectory corresponding to optimum parameter values and tracking performance between reference path and experimental output.*

Although, the initial path has been chosen as a straight-line in Cartesian space, the final optimized trajectories are completely different and provide fairly smooth manipulator trajectories.

The optimized manipulative task with different durations of motion is given in temporal trajectory position in Fig. 5-13. It is clearly seen from the figure that the end-effector of the robotic manipulator follows various trajectory motions for a given period of time in order to determine the optimal energy consumption.

In addition to this, tracking performance comparison was also done between theoretical optimized B-spline trajectories and experimental output of cubic trajectories with different durations of motion. The prescribed non-optimized end-effector's trajectory is also given. The corresponding path is shown in Fig. 5-14. It is seen from the figure that optimized theoretical and experimental output trajectories are completely different than the non-optimized manipulator trajectory and the experimental manipulator trajectory has an identical tracking performance with the theoretical one.

Fig. 5-15 shows the comparison between the theoretically simulated optimized velocity profiles (reference) and the experimentally recorded optimized velocity profiles (actual)

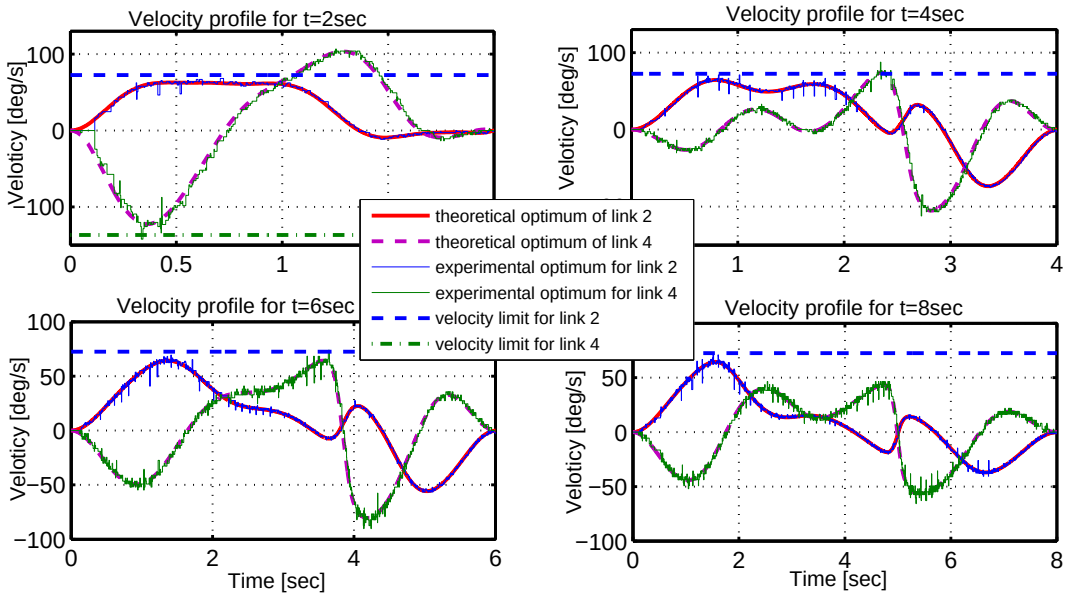

Figure 5-15: *Experimental comparison of speed with different duration of motion (Optimum motion).*

for link 2 and link 4, respectively. The straight dashed lines represent the velocity saturation for the link 2 and link 4, 73.5 deg/s and 134 deg/s, respectively. The velocities do not violate their velocity constraints. In this figure, experimental results of the velocity profiles support the simulation's outcomes, and they provide identical velocity profiles between the demand velocity and actual velocity. As expected, the faster motion of 2 seconds was found to deviate furthest from the demand velocity and was followed by the slower motions. Fig. 5-15 shows that optimized velocities are faster than the straight-line on EEF in Fig. 5-4 and also straight-line on joint motion in Fig. 5-9 velocities for the same period of times. The reason of this, as is seen from the Fig. 5-14 is that optimized motion trajectories are much longer than the non-optimized initial trajectories. In this case, in order to achieve the required optimum motion in a given time period, the robotic manipulator has to move quickly in order to achieve the motion successfully.

Figure. 5-16 demonstrates the theoretical comparison of variation of torque requirements with various durations of motion for the optimal trajectories. The results are plotted against the normalised time to allow a direct comparison to be made for different motion durations. The torque profiles in two second motion have bigger torque

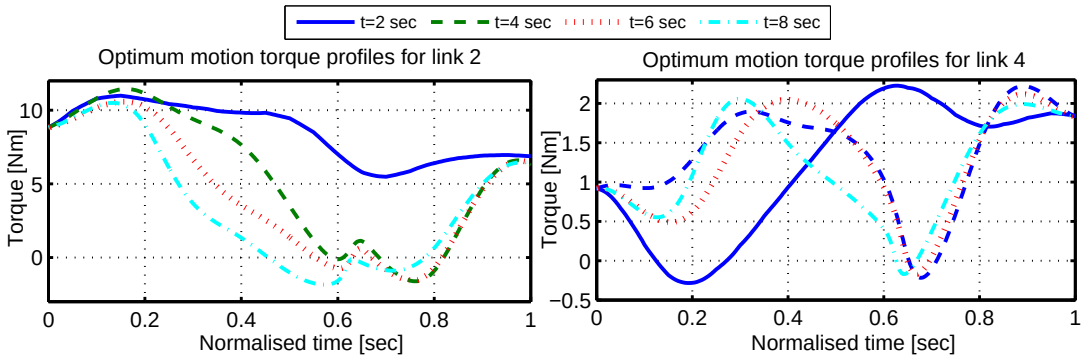

Figure 5-16: *Simulated planned trajectories of optimum path for various duration of motion with torque profiles (normalised time for direct comparisons between trajectories).*

magnitudes than the other profiles of the durations of motion. In 4, 6 and 8 second's durations of motion, the robotic manipulator lifted the second and the fourth links for a short period of time between 0 and 0.3 normalised second. This increased the cost value as it is seen from the Fig. 5-18 in 4, 6 and 8 second motions. After 0.3 normalised seconds, the robotic manipulator attempted to rotate the end-effector to the opposite direction in order to reach the final point. In this movement, links 2 and 4 of the robotic manipulator worked with the gravity as it is seen from the Fig. 5-13. In this case, as it is seen from the Fig. 5-18, the cost curve of the 4, 6 and 8 second motion profiles stayed almost in a horizontal position until the very end of the motion. This movement of the robotic manipulator provides low torque magnitude for the actuators, and it keeps the value of the cost function low. In addition to this, the torque magnitude of both actuators has been increased to keep the position on the final point, when approaching the end of the movement for the end-effector. This increased the torque requirements in the system as seen from the Fig. 5-18.

The corresponding current profiles are shown in Fig. 5-17 and are consistent with the velocity and torque profiles in Fig. 5-15 and Fig. 5-16, respectively. The current profile of link 2 for two second motion duration has a bigger current profile amongst the other. It is seen that the current profiles of link 4 are also consistent with the torque profile of link 4. Therefore, the results show a strong correlation between the theoretical and experimental outcomes.

Fig. 5-18 gives illustrative examples of the evolution of the cost functions for the

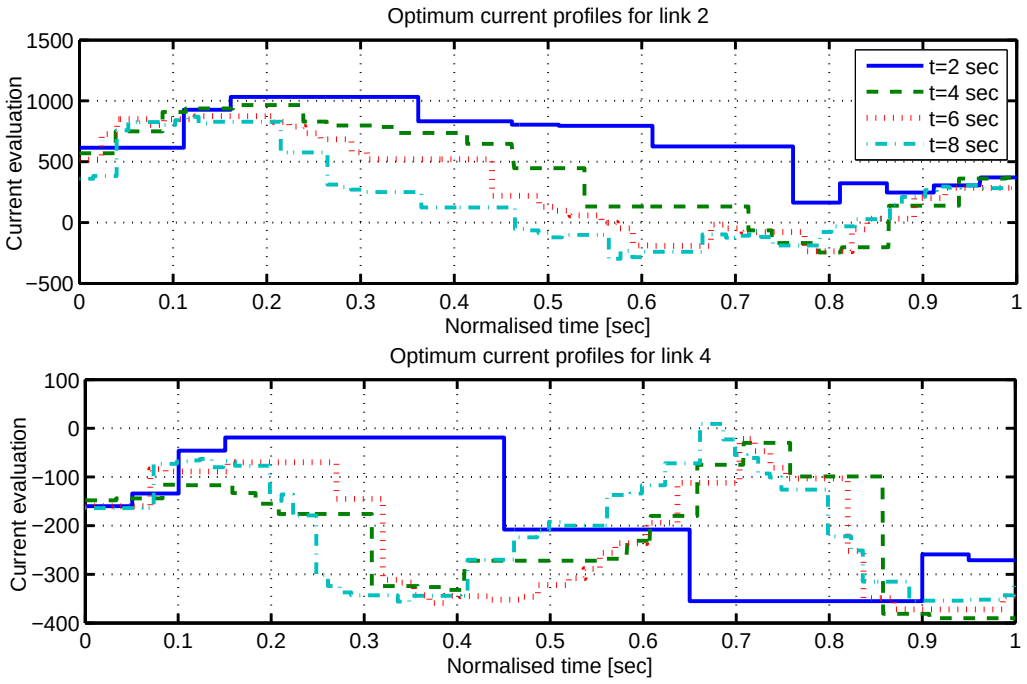

Figure 5-17: *Experimental comparison of optimum current with duration of motions.*

different types of trajectories with various time of duration of motion for the theoretical study. As we mentioned in section 4.1.3, optimization problems may consist of many local minima point during the trajectory optimization procedure. One can note and argue that, the global minimum point is not evident and clearly met, although the trajectory optimization process has given the result as a feasible optimum solution for the desired trajectory in Fig. 5-18. However, the variation of cost value is strongly constrained by physical kinematic and dynamic constraints for the actuated manipulator's links and given robotic manipulative job. Thus the optimum cost value can be taken

Time	Non-optimum(Simulation)	Optimum(Simulation)	Energy saving
	N^2m^2s	N^2m^2s	%
Cost(2s)	171	154	%10.2
Cost(4s)	299	203	%32
Cost(6s)	430	223	%48.1
Cost(8s)	561	248	%55.8

Table 5.3: *Cost values of the theoretical output of non-optimum and optimum motion with various duration of motion. Theoretical cost calculated from the required actuator torque to be applied at joint i.*

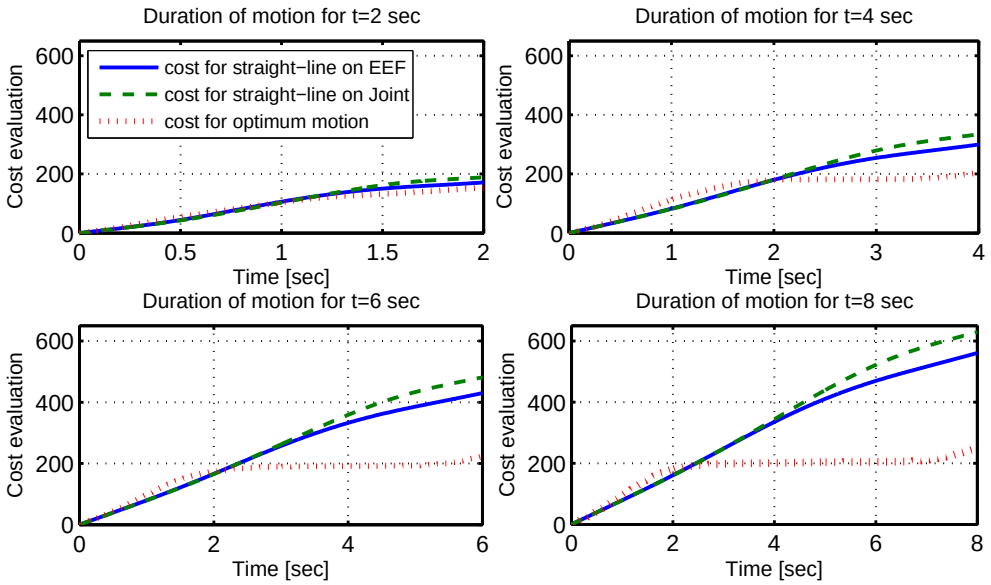

Figure 5-18: *Cost evaluation of theoretical simulation with the different types of trajectories and duration of motion.*

Time	Non-optimum(Katana)	Optimum(Katana)	Energy saving
	amp^2s	amp^2s	%
Cost(2s)	105	96	%8.71
Cost(4s)	135	103	%23.87
Cost(6s)	173	114	%34.50
Cost(8s)	212	120	%43.37

Table 5.4: *Cost values of the experimental output of non-optimum and optimum motion with various duration of motion. Experimental cost calculated provided current data for each axis.*

into account as a combination of kinematical and dynamical optimum characteristics of the system, as those related to the manipulator's link actions.

As demonstrated in Fig. 5-18, the duration of motion of the robotic trajectory has a crucial effect on the value of the cost function. Excessive growth of the cost values of the straight-line on EEF and also joint motion are shown to be due to increasing the duration of the motion from 2 seconds to 8 seconds for the required system. On the other hand, after optimization, the cost function is reduced significantly, the greatest improvement made was 55.8% in 8 seconds of motion. The corresponding cost values of the theoretical simulations are shown in Tab. 5.3. For the theoretical simulations, the cost calculated from the required actuator torque is to be applied at joint i. In order to

Time	Path	$\mathbf{r_1}$	r_2	r_3	r_4	$\mathbf{r_5}$	$\mathbf{r_6}$	r_7	r_8	r_9
	x_{eef}	**0**	0	0	0	**0**	**0**	0	0	0
	y_{eef}	**0**	0	0	0	**0.1940**	**0.3938**	0.6531	0.5235	0.7813
2(sec)	$Joint_{\theta_1}$	**0**	0	0	0	**0.3748**	**0.7609**	1.2618	1.0114	1.5096
	$Joint_{\theta_2}$	**0**	0	0	0	**0**	**0**	0	0	0
	x_{opt}	**0.2031**	-	-	-	**0**	**-0.0078**	-	-	-
	y_{opt}	**-0.1250**	-	-	-	**0.2409**	**0.6789**	-	-	-
	x_{eef}	**0**	0	0	0	**0**	**0**	**0**	0	0
	y_{eef}	**0**	0	0	0	**0.1940**	**0.3938**	0.6531	0.5235	0.7813
4(sec)	$Joint_{\theta_1}$	**0**	0	0	0	**0.3748**	**0.7609**	1.2618	1.0114	1.5096
	$Joint_{\theta_2}$	**0**	0	0	0	**0**	**0**	0	0	0
	x_{opt}	**-0.1730**	-	-	-	**-0.1698**	**-0.6362**	-	-	-
	y_{opt}	**-0.5155**	-	-	-	**0.7808**	**1.0338**	-	-	-
	x_{eef}	**0**	0	0	0	**0**	**0**	0	0	0
	y_{eef}	**0**	0	0	0	**0.1940**	**0.3938**	0.6531	0.5235	0.7813
6(sec)	$Joint_{\theta_1}$	**0**	0	0	0	**0.3748**	**0.7609**	1.2618	1.0114	1.5096
	$Joint_{\theta_2}$	**0**	0	0	0	**0**	**0**	0	0	0
	x_{opt}	**0.0770**	-	-	-	**-0.1737**	**-0.7260**	-	-	-
	y_{opt}	**-0.5155**	-	-	-	**0.7964**	**1.0377**	-	-	-
	x_{eef}	**0**	0	0	0	**0**	**0**	0	0	0
	y_{eef}	**0**	0	0	0	**0.1940**	**0.3938**	0.6531	0.5235	0.7813
8(sec)	$Joint_{\theta_1}$	**0**	0	0	0	**0.3748**	**0.7609**	1.2618	1.0114	1.5096
	$Joint_{\theta_2}$	**0**	0	0	0	**0**	**0**	0	0	0
	x_{opt}	**0.0770**	-	-	-	**-0.4588**	**-0.5971**	-	-	-
	y_{opt}	**-0.8827**	-	-	-	**0.7935**	**1.0318**	-	-	-

Table 5.5: *B-spline parameters for different types of trajectories with varying time of duration of motion.*

calculate the cost value for the experimental system, the required motion of the robotic manipulator's trajectory is executed five times and the current of each manipulator's link is recorded and averaged. This process provides a value roughly proportional to the required torque value. Then this current is utilised in the cost equation (Eq. 4.5) in order to calculate the required cost value for the desired movement. The corresponding experimental cost values are shown in Tab. 5.4 and maximum energy reduction was 43.37 %, which corresponds to 8 seconds of motion.

The B-spline parameters of straight-line on EEF motion (x_{eef}, y_{eef}), straight-line on joint motion ($Joint_{\theta_1}$, $Joint_{\theta_2}$) and optimum motion (x_{opt}, y_{opt}) with various durations of motion are given in Tab. 5.5. In order to satisfy initial and desired final conditions, three control points (r_2, r_3 and r_4) were used to satisfy the initial conditions (position, its first and second derivatives) and the other three control points (r_7, r_8 and r_9) were used to satisfy the end conditions (position, its first and second derivatives). The

remaining three free control parameters r_1, r_5 and r_6 (printed in bold in Tab. 5.5) are optimized by the optimization algorithm.

5.4 Error Analysis

In industrial implementations, the control of robotic motion has to be done effectively to increase the productivity as well as prevent the damage in the robotic manipulator structure. This can be achieved by optimal motion, which can be broken into two sections:

- Motion planning: consists of generating the trajectory motion and its time law, providing the reference signal for the robotic manipulator's controller.
- Motion tracking: concerned with developing the tracking performance of the reference signal in the required trajectory.

If the industrial implementation consists of repetitive implementations, in this case, improving the tracking precision of the robotic manipulator is always desirable. As described in this chapter, the non-redundant robotic manipulator scheme was implemented on the second and fourth links of the Katana 450 robotic manipulator. Between the initial and final points of the trajectory, three different types of manipulator's trajectories were executed on the Katana manipulator such as straight line on end-effectors motion, straight line on joint motion and also optimum motion on Cartesian coordinate with different motion durations. All of the error results are being discussed in the next subsequent sections.

$$Abs_{error} = \sqrt{(x_{actual} - x_{demand})^2 + (y_{actual} - y_{demand})^2} \tag{5.2}$$

The most convenient way to examine the outcomes in terms of the deviations is to determine the difference between the desired trajectory and the actual trajectory as demonstrated in Fig. 5-19. The absolute deviation between the theoretical trajectory of the end-effector and actual trajectory can be computed as the distance between the two points at a particular time instance as shown in Eq. 5.2. In the next subsections,

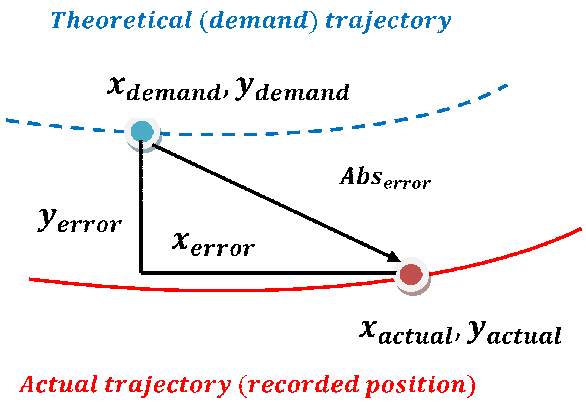

Figure 5-19: *Calculation of error at a particular time.*

different types of error will be discussed and shown in order to understand the error sources more clearly for the required trajectory motions.

5.4.1 Types of Errors

To better understand the reasons of error sources, a schematic diagram of the definition of different types of error sources is displayed as shown in Fig. 5-20 in order to identify the problem more clearly. Among the many types of error sources, two types of error sources are taken into account in order to investigate the quality of the actual trajectories as well as the affect on the result of the required system. Two types of error sources can be given as followed;

- Type 1 error: This type of error can be seen from the Fig. 5-20 and it can be named as "*Overall Tracking Error in Cartesian Space*". This type of error consists of a summation of three types of error sources, which are due to the cubic conversion for Katana axes, the encoder conversion for Katana robotic manipulator and also forward-kinematic conversion in order to get to the actual trajectory. It can also occur due to the flexibility of the robotic structure, the manipulator's control scheme and the uncertainty in the robotic manipulator's parameters.

- Type 2 error: The second type of error occurs between the desired B-spline trajectory in Cartesian space and the desired cubic polynomial trajectory in Cartesian

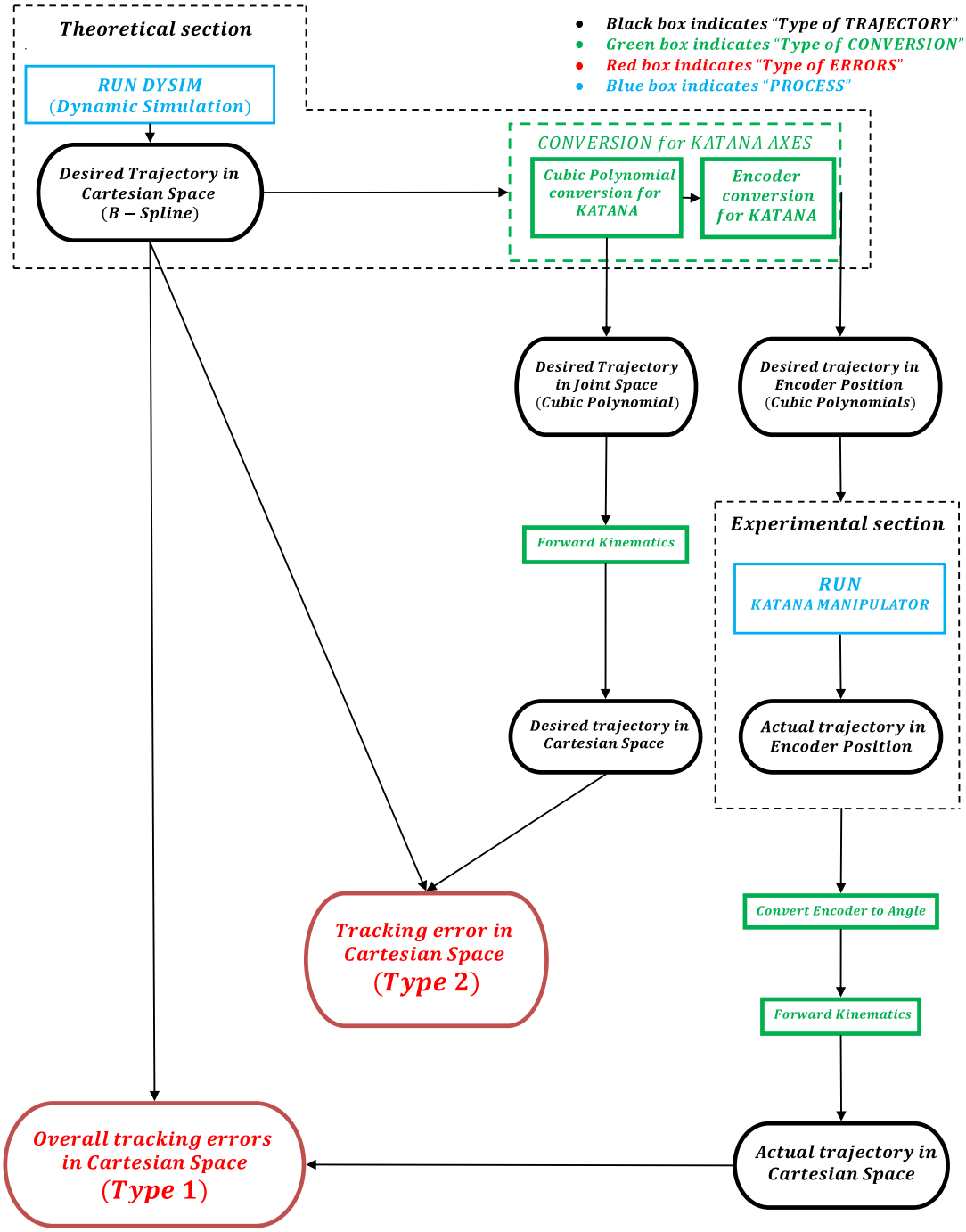

Figure 5-20: *Definition of different types of error sources.*

space and it can be named as "*Tracking error in Cartesian Space*" due to the cubic conversion and also forward kinematic conversion on the joint space trajectory.

5.4.2 Error in Straight-line on End-Effector Motion with Varying Duration of Motion

The straight-line EEF motion in Cartesian space is considered here. It is crucial to follow the particular trajectory accurately in order to provide successful implementations. Therefore, reducing the deviation of the end-effector is an important factor for any manipulative task. As it is seen from the Fig. 5-20, two types of error profiles were taken into account in order to demonstrate the quality of the generated manipulator's trajectories.

The path deviation plots for two different types of errors are demonstrated with respect to normalised time as given in Fig. 5-21. The experimental result supports the simulation result, that is, the absolute position deviation was determined to be proportional to the velocity profile as indicated by the Fig. 5-4. This means that the position deviation depends on the velocity profile, i.e., higher velocity profiles produce higher position deviations and vice-versa. For given velocity profiles in Fig. 5-4, the larger path deviation was determined high in 2 sec motion duration than the other durations of motion. It demonstrates that the path deviations in Cartesian space were reduced quite significantly by introducing a slower and smoother motion in the required system. It is seen from the Fig. 5-21 that the eight seconds motion duration has the smallest error profile for type 1 and 2 errors in the required trajectory motion.

When the type 2 error is considered in Fig. 5-21, there is a sharp and sudden drop in the curve profiles of the errors. As shown from that curve profile, this sudden drop in the curve occurs every 0.5 normalised time due to using 1 cubic polynomial for every 0.1 second in the desired trajectory. Normally, the shape of the curve profile should be smooth and the position at the connection point of each cubic segment is used as the initial values for the next segment of the cubic polynomial function as shown in Fig. 5-23(a). As previously mentioned in Chapter 3, in general, the ith cubic polynomial

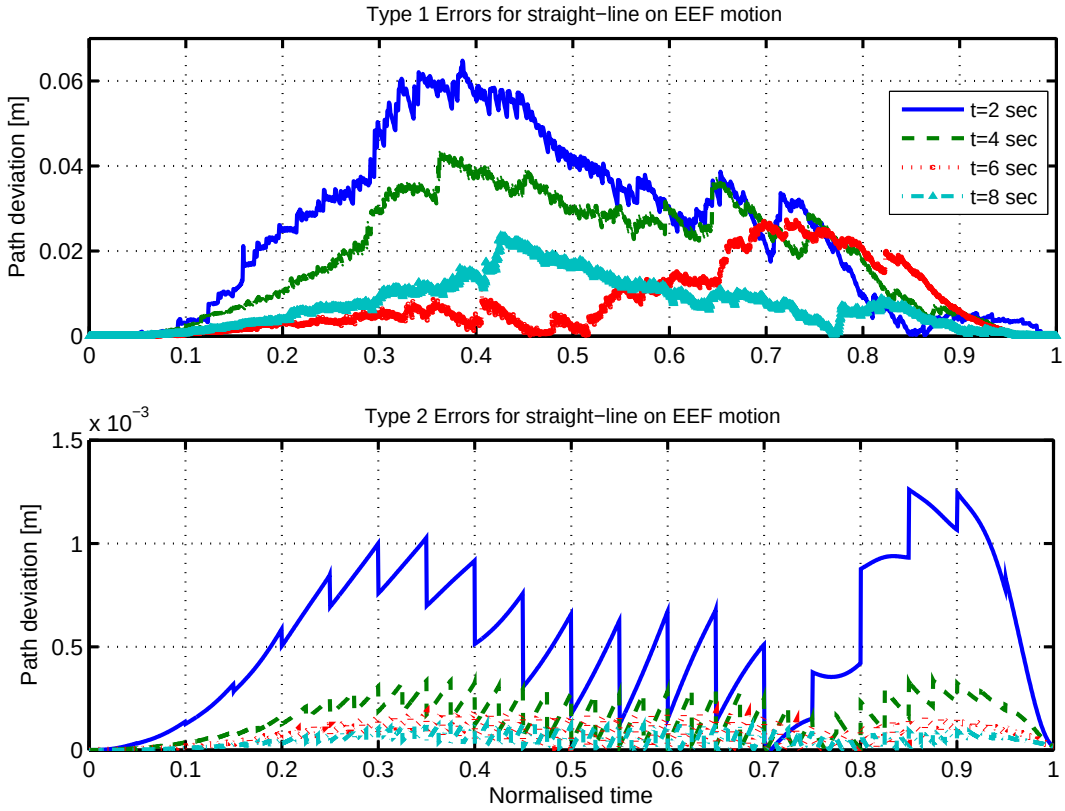

Figure 5-21: *(a) Type 1 Errors: Overall tracking errors in Cartesian space with varying duration of motion. (b) Type 2 Errors: Theoretical absolute error comparison for the simulated straight line trajectory (B-spline to cubic conversion) with varying duration of motion.*

function in terms of the parameter t, can be expresses in the form:

$$S_i(t) = a_{i,1} + a_{i,2}t + a_{i,3}t^2 + a_{i,4}t^3 \tag{5.3}$$

In this equation, $a_{i,1}$ belongs to any number before rounding off all values to the nearest integer values. However, in the next cubic segment, the previous value of the end point of a cubic polynomial is considered as the initial value for the next segment of the cubic polynomial function. Therefore, the error value at any connection point of the cubic segment normally should be zero as shown in Fig. 5-23(c). Even if there is a difference between the end point of the cubic segment and initial point of the next cubic segment after rounding off the all values into the usable integer values, we accept the end point of the cubic segment as an initial point of the next segment as shown in Fig. 5-23(b).

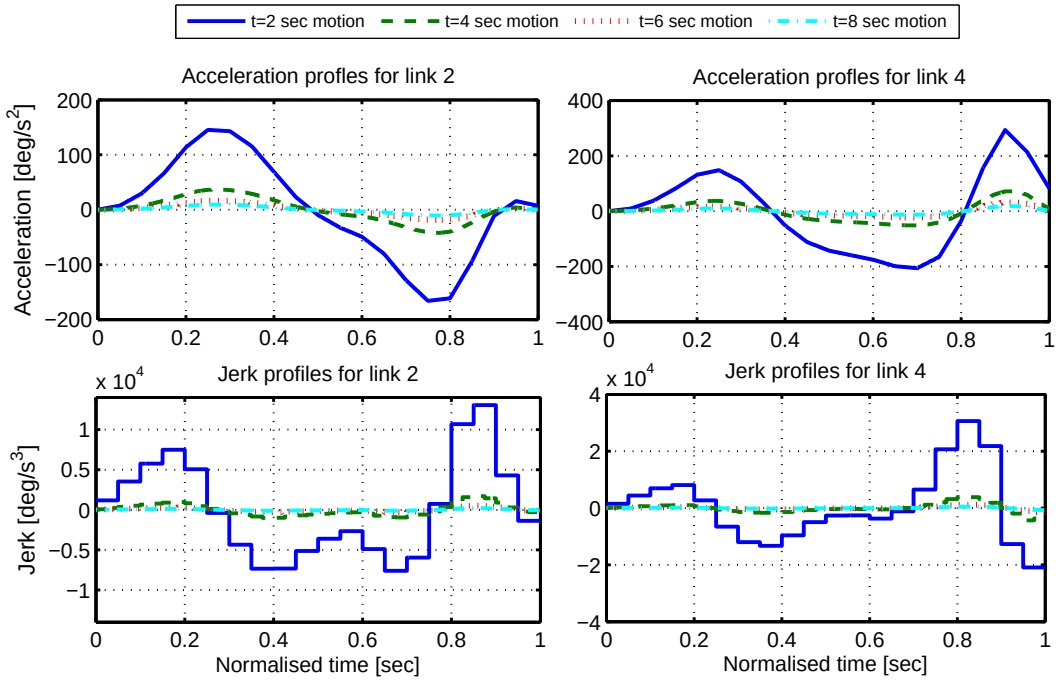

Figure 5-22: *Simulated planned trajectories of straight-line path for various duration of motion with acceleration and jerk profiles (normalised time for direct comparisons between trajectories).*

The cubic conversion program accepts the coefficient of the $a_{i,1}$ as the initial error value for the cubic trajectory. Although the rest of the equation $(a_{i,2}t + a_{i,3}t^2 + a_{i,4}t^3)$ of the cubic polynomial consists of small values during the generating of the cubic polynomials, the encoder position of the system can change largely. In this case, the error profile will increase due to largely changing the position of the manipulator and this results in an excessive increase in the deviation due to the small parameter profiles. This kind of error behaviour can be shown as in Fig. 5-23(d). This situation also helps us to describe the shape of the type of error 2 plots in Fig 5-21.

In order to have a better understanding on the error size and also quality of the actual trajectory, an error index is introduced in Tab. 5.6. The error table consists of scalar numbers, which represent the different types of error profiles over the required trajectory. With this error index at hand, a comparison between the desired and actual outcomes can be made more clearly. Two different error factors are introduced in the table; the first one represents the maximum absolute error values of the required trajectory due to the absolute errors for different types of error (as shown in Type 1

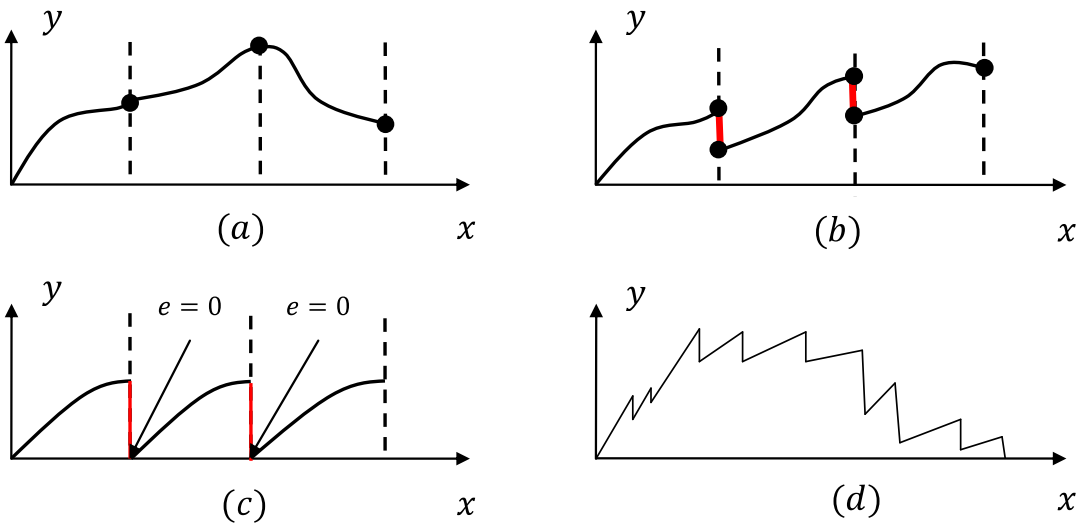

Figure 5-23: *Explanation of sudden drop in type 2 error plots.*

and Type 2) in the Cartesian space. The second factor represents the mean error which is the average of absolute error values.

The magnitude of the second type of errors is quite acceptable in the desired system, and two second motion has the largest deviation value of maximum and also average error profiles in the desired system due to the high velocity and acceleration profiles for this motion duration. The corresponding acceleration and jerk profiles are demonstrated in Fig. 5-22. As expected, when the required path velocities are increased in the demanded trajectory, the acceleration and jerk profiles are also increased inherently. It is seen from the Fig. 5-22 that the motion duration of 2 seconds has the largest acceleration and jerk profile for the required trajectory motion. On the other hand, the smallest tracking error value is obtained in the eight seconds motion duration which gives the best tracking performance in the average and also maximum error types val-

Type of Errors	2 sec	4 sec	6 sec	8 sec
Maximum error of type 1(m)	0.0648	0.0433	0.0273	0.0237
Maximum error of type 2(m)	0.00125	3.3828e-004	1.8721e-004	1.2683e-004
Mean error of type 1(m)	0.0250	0.0186	0.0079	0.0073
Mean error of type 2(m)	5.6836e-004	1.5318e-004	7.8169e-005	4.9896e-005

Table 5.6: *Error summary for straight-line motion with various duration of motion.*

ues. The error index shows that the maximum and also average errors of the required desired system were reduced quite significantly by introducing a slower and smoother motion in the desired system. Therefore, the end-effector of the robotic manipulator can track the trajectory more successfully by introducing the slower motion durations.

5.4.3 Error in Straight-line on Joint Trajectory with Varying Duration of Motion

The straight-line joint motion is considered in this section. The manipulator's control system of the joint motion will be acting on the robotic joints rather than on the end-effectors. Generally, the motions in joint space reach the end point with better precision and joint motion ensures smooth well-behaved end effector movement during the given task. As in section 5.4.2, the same initial and final positions are used and the duration of motion is also increased from 2 sec to 8 sec by a factor of 2. As is seen from the Fig. 5-24, link 2 has high acceleration and jerk profiles for two second

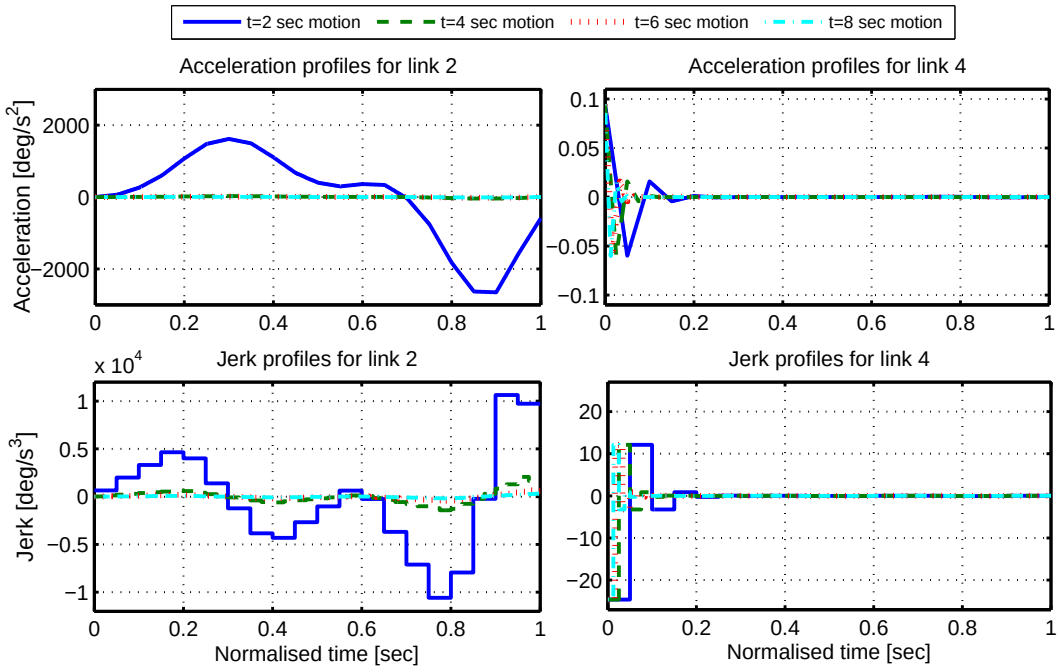

Figure 5-24: *Simulated planned trajectories of straight-line on joint motion path for various duration of motion with acceleration and jerk profiles (normalised time for direct comparisons between trajectories).*

motion duration. This type of motion can introduce large errors while the manipulator is performing its task. As is previously mentioned, link 4 did not change its position during the motion. Although the fifth-order B-spline functions provide zero velocity and acceleration profile for the initial and final point of the desired trajectory, third degree cubic polynomial functions do not guarantee the zero acceleration and jerk profile for the desired trajectory, and this can be seen from the Fig. 5-24.

By increasing the motion durations of the desired system, it results in slowing down the required motion, therefore, it provides a result with less positional error profile on the end-effector motion and a better accuracy performance in trajectory tracking. It was seen from the Fig. 5-9, the velocity profiles of the Katana for joint motion are slower than the velocity profiles of the Cartesian space motion in Fig. 5-4. That is,

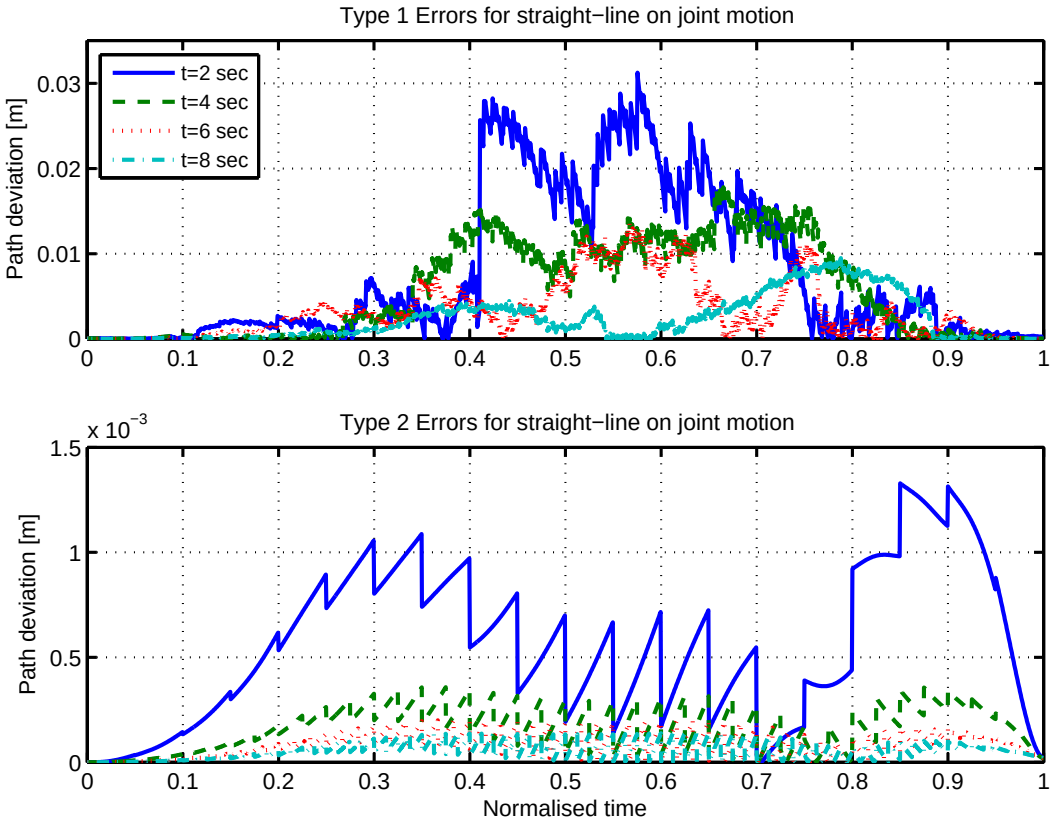

Figure 5-25: *(a) Type 1 Error: Overall tracking errors in Cartesian space with varying duration of motion. (b) Type 2 Error: Theoretical absolute error comparison for the simulated straight line on joint motion trajectory (B-spline to cubic conversion) with varying duration of motion.*

the velocities of the joint space motion corresponding to the Katana trajectories are smaller than those results from the straight-line EEF motion.

As shown in the Tab. 5.7, the maximum absolute and mean error of type 1 of the straight-line joint motion are much better than the errors in straight-line EEF motion in Tab. 5.6. From the Fig. 5-25, it can be seen that the type 2 error profile for joint motion are similar to the type 2 error profile for Cartesian motion in Fig. 5-21, but in a slightly smaller scale as shown in Tab. 5.7.

Type of Errors	2 sec	4 sec	6 sec	8 sec
Maximum error of type 1(m)	0.0312	0.0178	0.0136	0.010
Maximum error of type 2(m)	0.0013	3.5746e-004	2.0101e-004	1.4209e-004
Mean error of type 1(m)	0.0077	0.0058	0.0046	0.0033
Mean error of type 2(m)	5.5854e-004	1.6411e-004	8.4921e-005	5.4877e-005

Table 5.7: *Error summary for straight-line on joint trajectory with various duration of motion.*

5.4.4 Error in Optimum Motion with Varying Duration of Motion

Optimal velocity profiles for every duration of motion in Fig. 5-15 are faster than the non-optimized velocity profiles in section 5.4.2 and 5.4.3 for the same period of time. As it is seen from the Fig. 5-15, the optimum velocity profiles of link 2 of the Katana manipulator are close to its actuator limits for every duration of motion. In this case, as expected, the faster manipulator's motion will deviate the most from the demand position.

As mentioned before, higher velocity profiles produce higher acceleration and jerk values and vice-versa for any trajectory implementations. The acceleration and jerk profiles of optimum motion in Fig. 5-26 are quite different to the same profiles in non-optimized motion due to the velocity, and the manipulator's tracking behaviour for the desired task. As it can be seen from the Fig. 5-26, acceleration and jerk profiles of durations of motion of 4, 6 and 8 seconds have a sudden increase between 0.6 and 0.7 normalised time in link 2 and 4 of the robotic manipulator. This excessive growth of the curve can be shown to be due to attempt to rotate the end-effector position in the opposite direction in order to reach the final position as shown in Fig. 5-14. In Fig. 5-

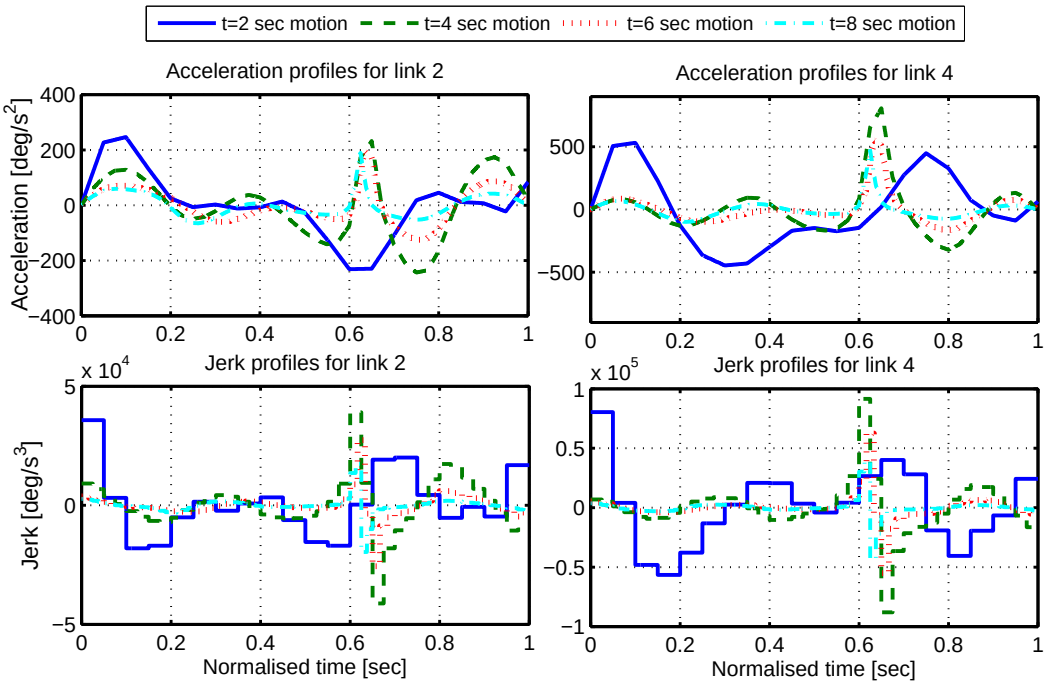

Figure 5-26: *Simulated planned trajectories of optimum motion for various duration of motion with acceleration and jerk profiles (normalised time for direct comparisons between trajectories).*

Type of Errors	2 sec	4 sec	6 sec	8 sec
Maximum error of type 1(m)	0.0252	0.0518	0.0598	0.0963
Maximum error of type 2(m)	0.00248	9.6957e-004	4.6707e-004	4.3184e-004
Mean error of type 1(m)	0.0048	0.0163	0.0181	0.0274
Mean error of type 2(m)	0.00106	2.8664e-004	1.3153e-004	1.0397e-004

Table 5.8: *Error summary for optimum trajectory motion with various duration of motion.*

27, the path deviation plots for two different types of error are shown with respect to normalised time.

Table. 5.8 displays the error index for each duration of motion for the optimum trajectory profile. The optimum type 1 error of the two second motion duration profile provides the best maximum absolute and average error values amongst the different type of trajectories such as straight-line EEF and joint space motion for two second motion duration. When the motion duration is increased from 2 sec to 8 sec by a factor of 2 for the straight-line on EEF and joint motion trajectories, the corresponding maximum and average error values are decreasing as it is seen from the Tab. 5.6 and

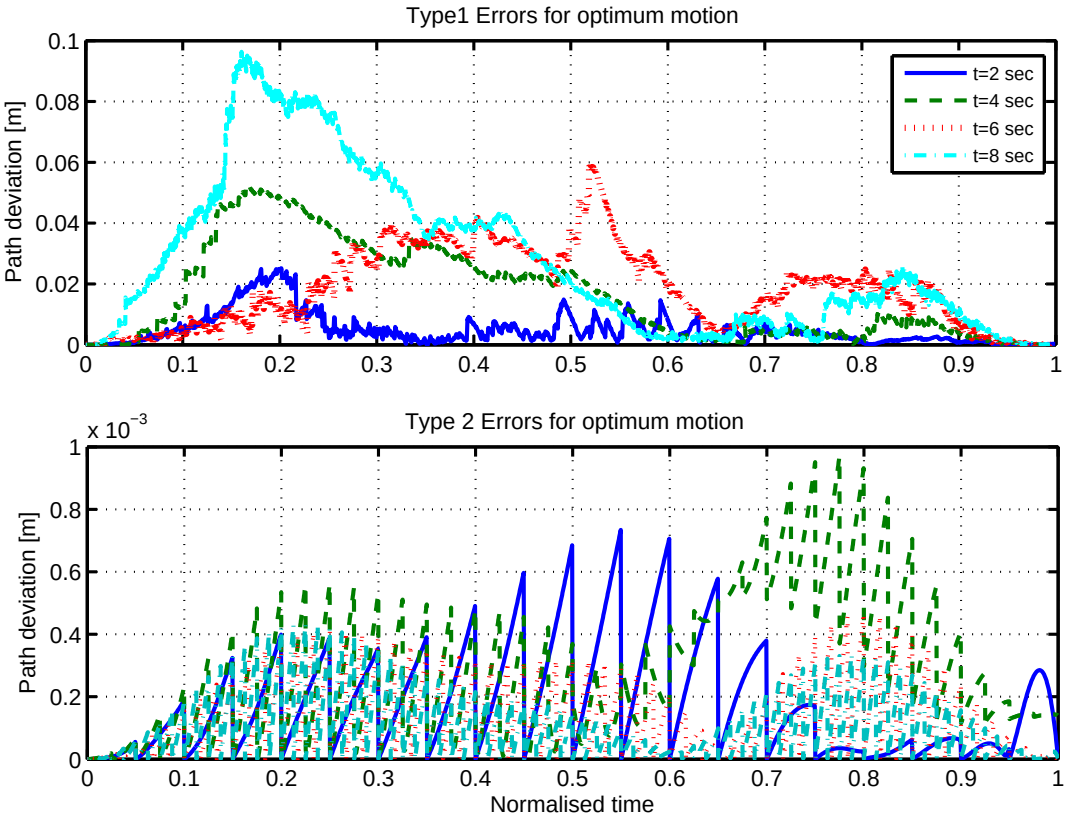

Figure 5-27: *(a) Type 1 Errors: Overall tracking errors in Cartesian space with varying duration of motion. (b) Type 2 Errors: Theoretical absolute error comparison for the simulated optimum motion trajectory (B-spline to cubic conversion) with varying duration of motion.*

Tab. 5.7, respectively. However, unlike the non-optimized trajectory's behaviour, values of the errors were increased in the optimum motion by introducing a slower motion for the desired system. This increment can be occurred due to the two different reasons. First, as shown in the Fig. 5-15, the optimum velocities require much higher velocity profiles than non-optimized trajectories. Hence, the larger velocity profiles provide larger tracking deviations in the desired system. Secondly, the end effector of the Katana manipulator in Fig. 5-14, has to travel much further than the non-optimized trajectories in section 5.4.2 and 5.4.3 for the same period of time in order to achieve the required motion on time. This may also result in cumulative error for the optimized trajectories.

5.5 Concluding Remarks

This chapter has dealt with the optimal trajectory planning using a proposed methodology for developing robotic trajectories that can achieve optimal minimum energy consumption for a point-to-point motion. Minimum energy consumption trajectories are computed by taking into account the main constraints imposed on the robotic kinematic and dynamic performance. The required trajectory is compatible with the robotic controller, because all of the kinematic and dynamic limitations of the robotic manipulator are handled effectively in the proposed optimization method before running the required trajectory experimentally.

An inverse dynamic analysis of a two degrees of freedom Katana 450 robotic manipulator is performed by using Lagrange dynamics and an in-house software package DYSIM. The DYSIM software is used to construct the equations of motion automatically for both forwards and inverse dynamic analysis of the desired system. Even two degrees of freedom motion is complicated, and the trajectory optimization problem is highly non-linear. This is due to the complexity of the dynamic model of the robotic manipulators that have to be verified during the trajectory optimization process and non-linearity of the cost function and kinematic and dynamic constraints. A uniform fifth-order B-spline function was used to define the end-effector in Cartesian space motion, and angles of the links in joint motion for the simulation study and hence the continuity of velocity and accelerations were guaranteed for the demand trajectory.

This was followed by the off-line motion planning method which converts B-spline trajectories into cubic polynomials due to Katana 450 input requirements. This off-line motion planning method has been successfully developed and used thoroughly in various experiments with various trajectory profiles of a two-link Katana 450 robotic manipulator based on link 2 and link 4. The trajectory planning of the robotic manipulator problem has been converted into a non-linear trajectory optimization problem by parameterezing the manipulator's end-effector and also joint temporal variations via cubic spline functions with each cubic equally distributed along a time scale. The parameters of the function are optimized utilising a multi-parametric trajectory opti-

mization algorithm. Actuator torques have been considered for the formulation of the cost function for the simulation study, and this utilizes an inverse dynamic analysis. For the experimental study, the current data was only provided from the Katana robotic manipulator. Therefore, the current was taken into account in order to calculate the cost values for the Katana motion.

Compared to the other trajectory optimization techniques, the proposed optimization method allows the system constraints to be handled within the cost function in order to avoid running the inverse dynamics when the constraints are not satisfied. Therefore, the complexity and computational effort of the optimization algorithm was reduced. It found that the proposed optimization algorithm worked successfully for any type of trajectory profiles.

The optimization results show that a significant improvement was achieved in the cost function for the simulation and experimental studies. However, it does not mean that the global minimum point has been found for the desired trajectory. On the other hand, the results indicate that the proposed optimization algorithm handles the kinematic and dynamic constraint effectively, and an important aspect of the trajectory optimization was implemented successfully.

Chapter 6 will be devoted to applying the same proposed optimization technique and also proposed algorithm to the redundant/hyper-redundant robotic manipulators. The reason of moving to this area is that the optimization problem is formulated in such a way that the same approach can be easily extended to handle the complex manipulator structures. Therefore, the next chapter will deal with the redundant/hyper-redundant robotic manipulators, which have more capability than non-redundant ones in many subjects. They play a significant role in increasing flexibility on the motion limits of joints, avoiding collisions with obstacles in the working space, and also the extra degrees of freedom of a redundant manipulator may provide a significant improvement in the result of the cost function. Moreover, implementation of the optimization technique and proposed alternative cost method will also be applied to the Katana robotic manipulator on the link 2, 3 and 4.

CHAPTER 6

OPTIMUM TRAJECTORY PLANNING FOR REDUNDANT/HYPER-REDUNDANT MANIPULATORS

In the previous chapter, minimum energy consumption trajectories for a point-to-point motion under kinematic and dynamic constraints were performed theoretically and also experimentally for a non-redundant robotic manipulator. Links 2 and 4 of the Katana 450 industrial robotic manipulator were utilised.

This chapter focuses on efficient control techniques for redundant/hyper-redundant manipulators to avoid the computational complexity associated with the redundant and hyper redundant manipulators with a large number of DOFs. The optimal trajectory planning for a redundant manipulator was proposed with fifth-order B-splines to represent Cartesian coordinates of the end-effectors trajectory ($f_x(t)$ and $f_y(t)$), and the relative angle of each redundant link. In order to perform the optimization method experimentally, the fifth-order B-spline trajectory was converted to the cubic polynomials (see more detail in section 3.3.1) in order to meet the Katana input requirements. The actuator torques have been considered for the formulation of the cost function for the simulation study, the current values are used to calculate the cost function in the experimental studies. Calculation of the cost function and dynamic of the mechanism is carried out by using an inverse dynamic analysis.

A three-link Katana robotic manipulator based on links 2, 3 and 4 is used for the-

oretical and also experimental studies to demonstrate the effectiveness of the proposed virtual link concept. In addition to this, the latter part of this chapter deals with the design and implementation of an 8-links hyper-redundant manipulator, which is utilised to demonstrate the validity of the proposed control technique for a larger number of DOFs manipulator. Through simulations and experiments, the energy minimization of the proposed method is also verified. The path deviation along the desired manipulator trajectories was also compared in order to determine the tracking accuracy for the desired manipulative task with various durations of motion.

The novelty of work claimed in this chapter includes [145]:

- The system constraints are handled within the cost function to avoid running the inverse dynamics when the constraints are not satisfied.
- The control scheme relies on the definition of a virtual link concept, where all the redundant links are acting as a single link during the motion. Hence, this method makes controlling these links easier, reducing the control complexity of the redundant/hyper-redundant manipulators. This process is applicable to hyper-redundant manipulators with a large number of links.
- A virtual link concept eliminates physically impossible configurations before running the inverse dynamic model. Therefore, this control algorithm prevents inverse dynamic solution failure (even if the manipulator is within the workspace) during the optimization process.
- Control forces are also calculated for each link.

6.1 Formulation of the Optimization Problem

To achieve a desired trajectory for any manipulative task, only second derivatives of the position profile are required in the inverse dynamic modelling as discussed in Ch. 3, in the following form [8]:

$$\ddot{\mathbf{y}} = \mathbf{C}_1 \ddot{\mathbf{q}} - \mathbf{C}_2 \tag{6.1}$$

where $\mathbf{C_1}$ is of the dimensions $K \times N$. The value of K is equal to or less than the degrees of freedom (M) of the system. K independent inputs have to be selected in order to achieve the desired motion $\ddot{\mathbf{y}}$.

Actuated systems can be divided into three sections [8];

- Fully actuated systems ($K = M$)
- Underactuated systems ($K < M$)
- Redundant systems ($K > M$)

In a fully actuated system, the number of control inputs K is equal to the degrees of freedom M of the system. In the underactuated case, the number of the control inputs K is less than the degrees of freedom M of the system. However, in the redundant case, a system may have more control inputs than required in order to control a specified desired motion. In this case, the inverse dynamic equations consist of more unknowns than the number of equations [8].

For instance, consider the n-links redundant robotic manipulators shown in Fig. 6-1, where relative joint angles are denoted by θ_i and link lengths are denoted by l_i for the ith link. The manipulator task consists of transporting a load mass, m_{load}, from an initial point, $\mathbf{P_{initial}}$, to a destination, $\mathbf{P_{final}}$, in Cartesian space. This system can be modeled by using the Lagrange's equation of motion as described in Ch. 3.

Dysim is utilized to automatically develop a dynamic model of the system. Cartesian coordinates of the centre of gravity and the relative angles of each link plus the Cartesian coordinates of the load are selected as generalized coordinates, i.e. a total of $(3n + 2)$ generalized coordinates for the n degrees of freedom system. Dysim automatically develops $(2n + 2)$ constraint equations and a constraint Jacobian matrix $\mathbf{F}$ [145].

For the formulation of the inverse dynamic model, the parametric desired motion for this n degrees of freedom system is specified by using the Cartesian coordinates of the end-effector (f_x and f_y), and the first $(n - 2)$ relative angles of the redundant manipulator ($\theta_i, i = 1 \cdots n-2$). The last two relative angles (θ_{n-1} and θ_n) are selected

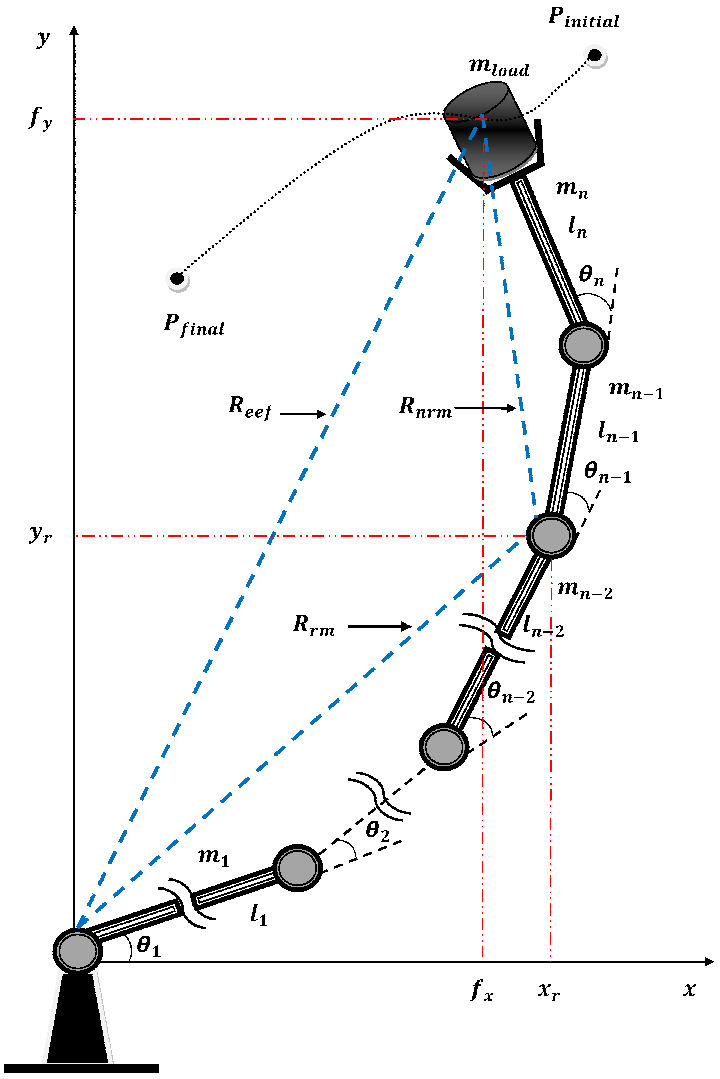

Figure 6-1: *Schematic view of a redundant manipulator.*

as dependant coordinates. The n control input locations are selected to be the relative angles of all n links (i.e. the joint actuator torques) [145].

A uniform fifth-order B-spline function was also used to define the desired trajectory for the redundant/hyper-redundant manipulators study. As it has been discussed in section 4.1.1, in order to satisfy initial and desired final conditions, three control points (r_2, r_3 and r_4) were used to satisfy the initial conditions (position, its first and second derivatives) and the other three control points (r_7, r_8 and r_9) were used to satisfy the end conditions (position, its first and second derivatives). The remaining three free control points (r_1, r_5 and r_6) are optimized by the optimization algorithm [145].

As a result of this, 6 free parameters for the trajectory of the end-effector in Cartesian coordinates ($f_x(t)$ and $f_y(t)$), and 4 free parameters for the relative angle of each redundant links are used by the optimization algorithm. In the case of redundant manipulators, the end position of the redundant links is also to be optimized and not known in advance. In order to start the optimization algorithm, initial values of the free parameters have to be specified. Arbitrary selection of the free parameters is used for the initial trajectory in the optimization algorithm for the redundant/hyper-redundant manipulators [145]. The velocity and acceleration profiles are zero at the initial and final positions. As mentioned in subsection 3.3.1, the desired fifth-order B-spline trajectory will also be converted to cubic polynomials for the redundant case in order to provide the required input parameters for the Katana robotic manipulator axes.

6.1.1 Proposed Virtual Link Concept (Redundant Links Reach)

Another requirement for the motion defined by the optimization parameters to be realizable is that the distance between the end of the last redundant link (point at x_r, y_r) and the end-effector (point at f_x, f_y) must be less than or equal to $(l_{n-1}+l_n)$. This is equivalent to replacing the redundant links by a single virtual link R_{rm} as shown in Fig. 6-1. At each step of the movement, the position of the redundant/hyper-redundant robotic manipulators has to satisfy the following constraint, which can be formulated as follows [145]:

$$R_{nrm} = \max \sqrt{(f_x - x_r)^2 + (f_y - y_r)^2} \leq l_{n-1} + l_n \tag{6.2}$$

where x_r and y_r can be calculated from the optimization parameters as follows:

$$x_r = \sum_{i=1}^{n-2} l_i \cos\theta_i \quad \text{and} \quad y_r = \sum_{i=1}^{n-2} l_i \sin\theta_i \tag{6.3}$$

The most important advantage of this proposed method is that whatever the length of the redundant links, it will always be acting as a virtual link (as a single link) (R_{rm}). Figure. 6-2 demonstrates the simplified view of Fig. 6-1, and it can be seen from

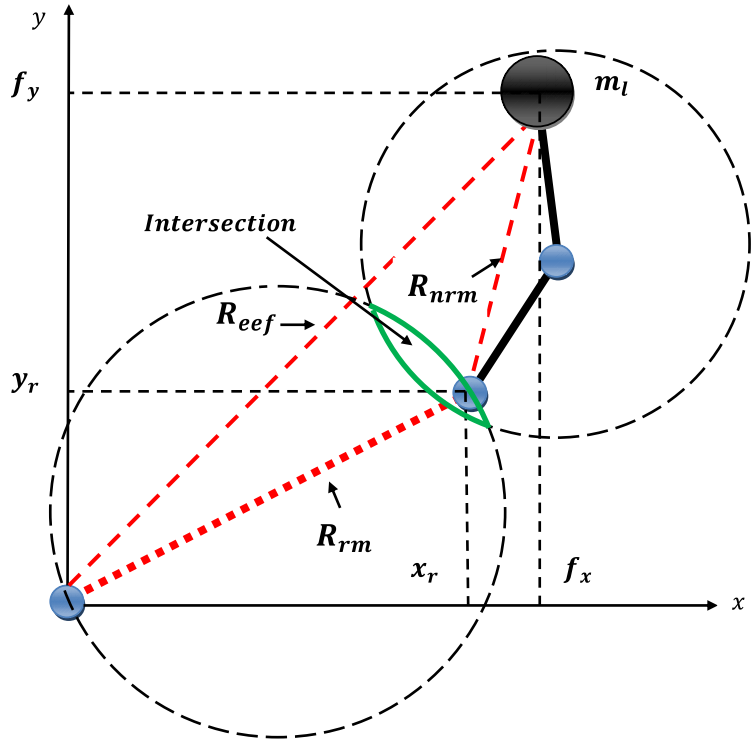

Figure 6-2: *Simple model of the virtual link.*

the Fig. 6-2, the control algorithm in Eq. 6.2 enables intersection of the two circles. This intersection point of two circles will be satisfied at each step of the movement. Hence, redundant links position on the workspace is guaranteed by a virtual link (R_{rm}) and computational complexity of the redundant/hyper-redundant links are significantly reduced [145].

As it is seen from the Fig. 6-1, except the last two non-redundant links, all the other links have redundancy. Redundant robotic manipulators may consist of any number of DOFs and any length. In this case, each of the redundant links has an infinite number of solutions for the desired end-effector position. This situation can cause many problems when executing the optimization, such as the complexity in computational, long computations time, and difficulty in finding the optimal solution between the infinite numbers of solutions. However, this proposed algorithm has the capability to control a large number of DOFs manipulator while reducing the energy consumption in the optimization algorithm. In addition to this, the proposed method can easily be extended to 3D cases. Hence, the complexity of handling a large number of DOFs is

significantly reduced [145].

6.2 Constraint

In addition to the system and task constraints which were discussed in section 4.1.4, it is important to ensure that the parametric trajectory functions generated by the optimization algorithm result in a realizable motion within the workspace of the manipulator. Otherwise, the inverse dynamic simulation will fail to run during the cost function calculation and hence the optimization will fail. For example during the motion, the Cartesian coordinates of the end-effector may not be achieved by the angles of the redundant links (i.e. the first $n-2$ links). Therefore, the following additional constraint equation is added to the existing constraints to prevent the inverse dynamic simulation failing during the desired motion [145].

6.2.1 End-effector Reach

This non-linear constraint ensures the end-effector is on a reachable point in Cartesian coordinates. In this criterion, the distance between the origin and end-effector must be less than the sum of all of the link lengths. The following criterion must satisfy the end-effectors reachable point R_{eef} in Cartesian coordinates [145]:

$$R_{eef} = \max \sqrt{(f_x^2(t) + f_y^2(t))} \leq \sum_{i=1}^{n} l_i \tag{6.4}$$

for $0 \leq t \leq T$. Since f_x and f_y are generated from the optimization parameters, this constraint can be checked before calling the inverse dynamics.

6.3 Optimum Trajectory Planning for 3-Links Redundant Manipulator on Cartesian Coordinates

This section deals with the optimal trajectory planning of redundant/hyper-redundant robotic manipulators based on the criterion of torque minimization on a 3-links redundant robotic manipulator (based on links 2, 3 and 4 of the Katana 450 robotic

manipulator) for the theoretical and experimental studies. An inverse dynamic model of the 3-links Katana 450 is considered for the proposed control method by using the Lagrangian's equation of motion which is carried out by the program of DYSIM.

The cost function is given by the motion duration integral of the squared required actuator torques as given in section 4.1.2 for the simulation study. The optimization routine takes into account inverse dynamic analysis which requires the acceleration profile of the motion as given by fifth-order B-spline functions. The continuity of velocity and acceleration as well as zero velocity and acceleration at the start and end positions of the motion are guaranteed for the simulation study. All the theoretical

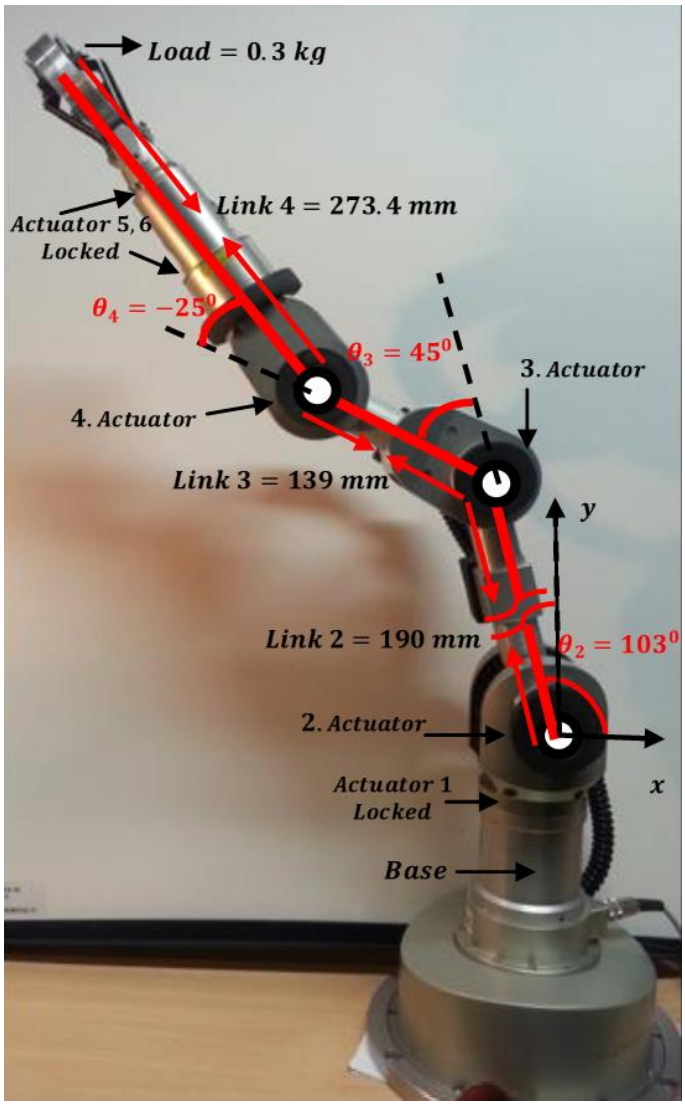

Figure 6-3: *3-links Katana model based on link-2, link-3 and link-4 (rotary joints) with one redundancy in link-2. The base of the manipulator, link-5 and link-6 are locked for this scheme.*

simulations for 3-link redundant robotic manipulators are also executed experimentally on the Katana 450 robotic manipulator. The experimental data from the Katana 450 axes was recorded as raw data such as encoder position, encoder velocity and encoder time, and then this recorded data is analysed and converted to angles in degrees to compare with the demand and actual trajectories. As mentioned in the previous chapter, acceleration and jerk profiles will not be implemented for the experimental results due to the limitations of the AxNI Scope of AxNI software.

To compare and analyse the performance of the 3-link redundant robotic manipulator, the different duration of motion of the non-optimal and optimal trajectories share the same initial and final value of the end-effector position for the simulation and experimental studies. For the 3-link redundant manipulator scheme, the manipulator task consists of transporting a load mass of 0.3 kg from an initial point at $(x_i = -0.3095, \quad y_i = 0.4881)$ m to a destination at $(x_f = 0.1405, \quad y_f = 0.4381)$ m in Cartesian space as shown in Fig. 6-3. This is done by considering the kinematic and dynamic constraints imposed on the redundant robotic manipulator. It can be seen from the Fig. 6-3 that the first link of the manipulator has redundancy, and the rest of the links are non-redundant. For the 3-link redundant scheme, three motors control the motion. Also, kinematic and dynamic constraints expressed by means of upper and lower limits on angle, velocity, acceleration and input torque are taken into account and shown in Tab. 6.1. The viscous friction effects of the actuators are also taken into account with coefficients of friction of 1.8 Nms/rad, 0.4 Nms/rad and 0.39 Nms/rad for links 2, 3 and 4, respectively. The gears are also taken into account with the gear ratios as given in Tab. 3.3. The mass centre of gravity of the links are identified as shown in Tab. 3.1 and the load mass is $m_{load} = 0.3$ kg and is attached at the end of

Limits	**Link 2**	**Link 3**	**Link 4**
Angle (deg)	+102°/-30°	+/-122.5°	+/-112°
Max velocity (deg/s)	72.52	73.53	136.8
Maximum acceleration (deg/s^3)	2321	2353	4378
Torque (Nm)	17	13	9

Table 6.1: *Limiting parameters used on Katana 450 6M industrial robotic manipulator.*

the last link of the manipulator. The total duration of motion is varied from 4 sec to 10 sec by increments of 2 sec.

The redundant robotic manipulator has three degrees of freedom, the Dysim program selects 11 generalised coordinates (three for each links and two for the load) for the robotic manipulator as follows:

$$q = [x_1, y_1, \theta_1, x_2, y_2, \theta_2, x_3, y_3, \theta_3, x_L, y_L] \tag{6.5}$$

where x_i, y_i and θ_i are the Cartesian coordinates of centre of gravity, and the joint angle, respectively, for link i. The system consists of 20 constraints, 23 variables. The Dysim program automatically develops the Lagrangian function and the dynamic equations of motion including constraint equations and differential-algebraic equations. The initial conditions of the dependent coordinates based on the user defined initial position conditions of the user selected three independent coordinates, angle of θ_1, θ_2 and θ_3 are also automatically calculated by the Dysim program. In this case, θ_1, x_L and y_L were selected as the motion defining independent variables.

In the non-redundant case, to start the optimization algorithm the initial path was selected as a straight-line trajectory in the Cartesian space between the initial and final point of the desired trajectory. However, in the redundant case, the initial values of the free parameters are selected randomly, but close enough to the straight-line trajectory. It is important to note that if the robotic manipulator has a redundancy, the outcome of the constrained optimization procedure can be more sensitive to a given initial free parameter due to the numerical complexity of the redundant links. This is because, the redundant link has the capability of moving the joints in infinite ways for the same specified end-effector motion. Therefore, the smallest change in the starting conditions can result in a major difference in the outcome of the optimization algorithm.

6.3.1 Optimization Results of Redundant Manipulator

The prescribed manipulative task and the obtained optimum theoretical and experimental paths are shown in Fig. 6-4. As it is seen from the figure, the final optimized

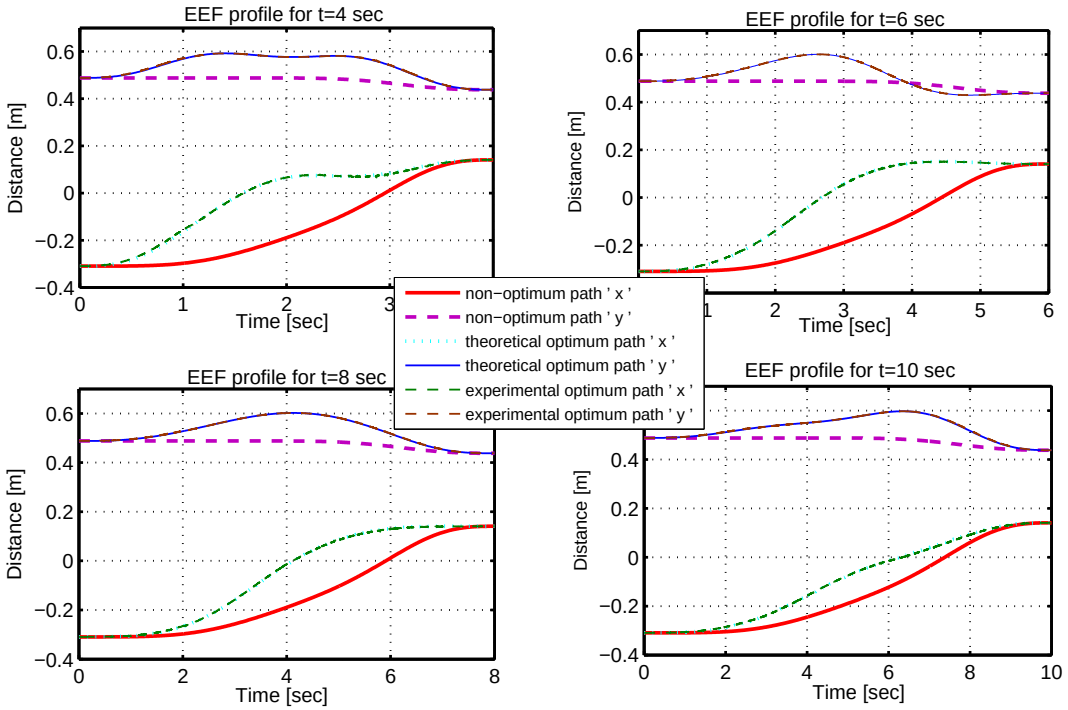

Figure 6-4: *Comparison of theoretical non-optimum, optimum position and also experimental optimum positions of the trajectories.*

path is totally different from the initial non-optimized path, and the experimental optimized trajectory is consistent with the optimized theoretical trajectory. Figure 6-5 shows the corresponding optimized manipulative task with varying durations of motion in temporal trajectory position. The temporal positions in this figure clearly show us the manipulator's behaviour during the simulations. It can be seen from the figure that the robotic manipulator has followed different paths for each duration of motion without violating the kinematic and dynamic constraint conditions. In addition to this, tracking performance comparison was also made between theoretical optimized B-spline trajectories and experimental optimized output of cubic polynomial trajectories with varying durations of motion. The corresponding path is shown in Fig. 6-6 and is consistent with the end-effector trajectory of the temporal positions of the robotic manipulator as shown in Fig. 6-5.

Singularity avoidance of a robotic manipulator is also a main subject for robotic design and trajectory planning. Basically, singularity avoidance means keeping a current robotic configuration away from a set of singular configurations [161]. Physically,

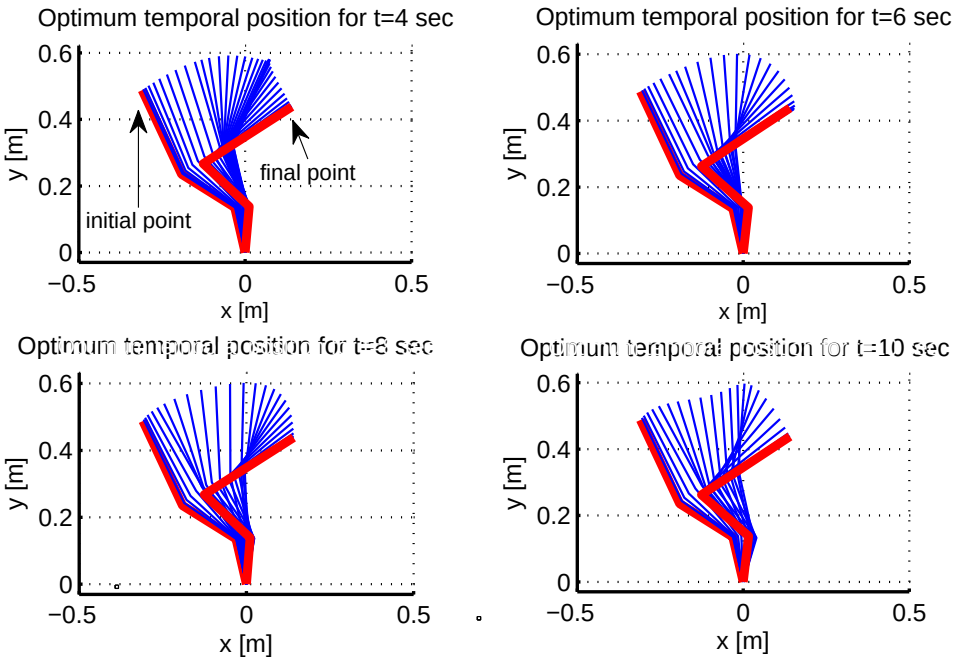

Figure 6-5: *Temporal position of theoretical optimized trajectory of 3 link redundant robotic manipulator with varying durations of motion (DYSIM output).*

a singular configuration would cause a sudden change in the DOF of the robotic manipulator structure and then cause a loss of a controllability of the robotic manipulator's motion at, or near, a singular position. In this singular point of the trajectory, the ability of the manipulator is very limited. Depending on a trajectory to follow, singularity may occur many times in the desired system. It is crucial to determine or avoid the singular configuration of the robotic manipulator to improve system performance [161]. However, in the case of redundancy, this system may be more capable of preventing the singularity point due to having an infinite number of solutions to the joint variables for the same specified end-effector's position [82].

Figure 6-7 shows the output determinant of the augmented matrix for varying durations of optimum motion of redundant trajectories. This determinant is a good candidate to indicate a singularity point of the resulting trajectories. Output determinant has been presented by **A** matrix as shown in Eq. (3.7).

Simulation results of the output determinants presented in Fig. 6-7 show that while approaching a singularity point at $t = 1.1$ sec, $t = 1.55$ sec, $t = 2$ sec, $t = 2.65$ sec,

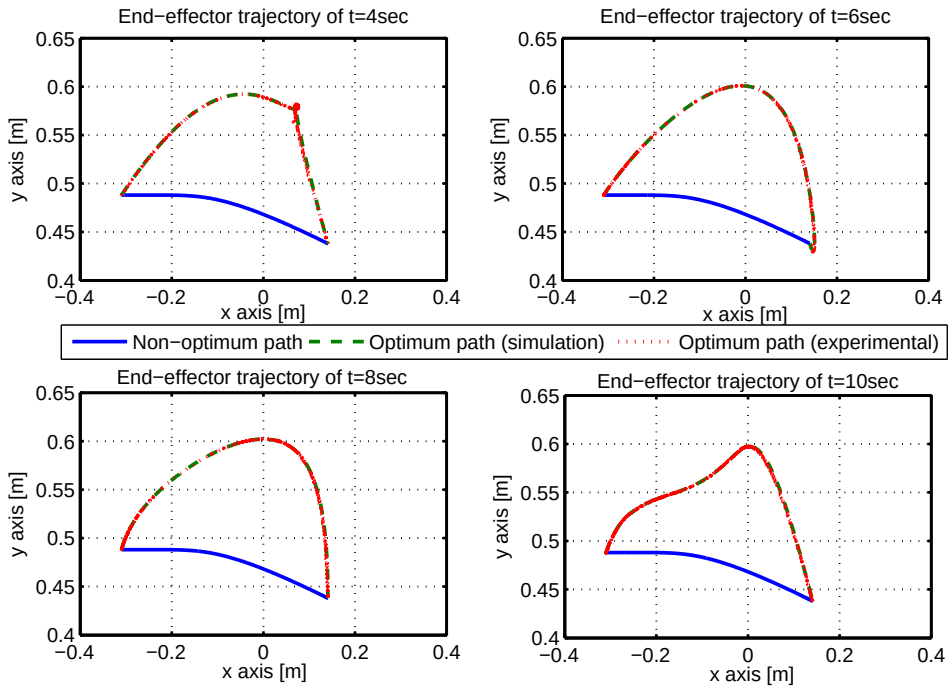

Figure 6-6: *Initial non-optimum path, motion trajectory corresponding to optimum parameters values and tracking performance between reference path and experimental output.*

for the durations of motion of $t = 2$ sec, $t = 4$ sec, $t = 6$ sec, $t = 8$ second of duration of motion, respectively, the determinants of the redundant manipulator become zero indicating singularity configurations. At these singularity points, relative angles of link 4 to the link 3 of the robotic manipulators are almost zero degrees as indicated in the red colour link as shown in Fig. 6-8.

However, the current robotic configurations allow the robotic manipulator to move away from singularity point and it quickly recovers from such a situation without stopping or/and stucking in a singular point during the desired motion. Although a smoothness of passing through singularity points is not guaranteed completely, all of the determinants changed their signs around the singularity points, and motion continued.

Figure 6-9 presents the comparison between the theoretically simulated optimized velocity profiles (reference) and the experimentally recorded optimized velocity profiles (actual) for links 2, 3 and 4, respectively. The velocities do not violate the velocity constraints. As expected, experimental results of the velocities support the simulation results, and identical velocity profiles between the demand and actual velocity have

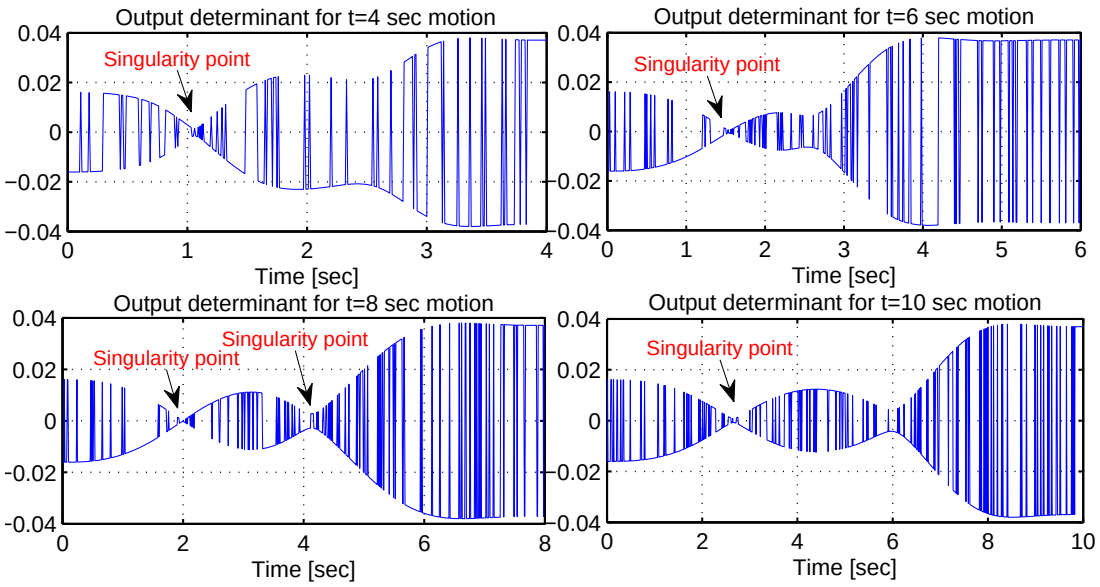

Figure 6-7: *Output determinants of the 3-link redundant robotic manipulator with varying duration of motions.*

been observed.

As it can be seen from the velocity profile for 4 second in Fig. 6-9, the link 2 is the slowest link amongst the other two links. In this duration of the motion, the robotic manipulator attempts to move the link 3 and link 4 away rapidly from the initial point, and this movement results in a high velocity profile for the link 3 and link 4 during that motion. While approaching a singular point at 1.1 seconds (as shown in output determinant for 4 second motion in Fig. 6-7), there is a sudden and sharp decline in the velocity profiles of link 3 and link 4. At a singular point at 1.1 seconds, the speeds of the link 3 and link 4 reach almost zero velocity for this duration of motion. In addition to this, link 2 becomes perpendicular to the x coordinate as shown in "temporal position for 4 sec profile (a)" as shown in Fig 6-8. This duration of motion is also demonstrated in the actuator torque profiles as in Figure 6-10, which shows the theoretical comparison of variation of torque requirements with various durations of motion for the optimal redundant trajectories. As a result of this, a sudden decrease in the torque profile of link 2 is observed at 0.26 normalised time as shown in Fig. 6-10 and it reaches zero torque magnitude at this duration of motion. This torque profile is consistent with the configuration of the link 2 in a "temporal position for 4 sec profile (a)" as shown

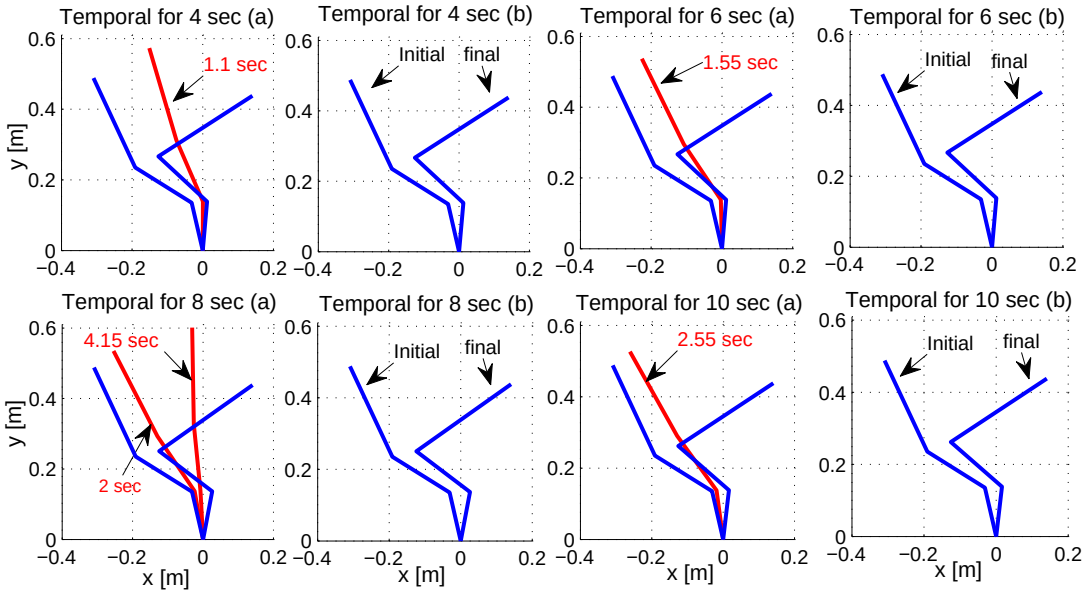

Figure 6-8: *The corresponding simulated link positions for the singularity points of the output determinants with varying durations of motion.*

in Fig 6-8. After 0.4 normalised seconds, link 2 is taking the position almost vertical to the x axis, therefore it has an almost zero torque magnitude until the end of the simulation.

Furthermore, the duration of the motion of 6 second has almost the same velocity and determinant profiles as the duration of motion of 4 second during the given task. As is seen from the velocity profile for 6 seconds of motion in Fig. 6-9, they are inherently slower than the velocity profile of four second. In the duration of motion of 6 second, the velocity profile of the link 3 and link 4 also have a sudden and sharp decline before they reach the 1.55 seconds of simulation time. Similar to the velocity profile of 4 second motion, in the duration of motion at 1.55 second, the velocity profile of the link 3 and link 4 becomes almost zero and this type of movement is consistent with the determinant profile of duration of motion of 6 second as shown in Fig. 6-7. Similarly, in this singular configuration of the motion, link 4 is in the zero angle position relative to the link 3. In this duration of motion, the required torque profile is decreasing, and the necessary torque profile of 6 second motion is clearly less than the duration of motion of 4 second profile. Correspondingly, link 2 also takes the position as perpendicular to the x axis between the 0.4 and 1 normalised second of the torque profile of 6 second

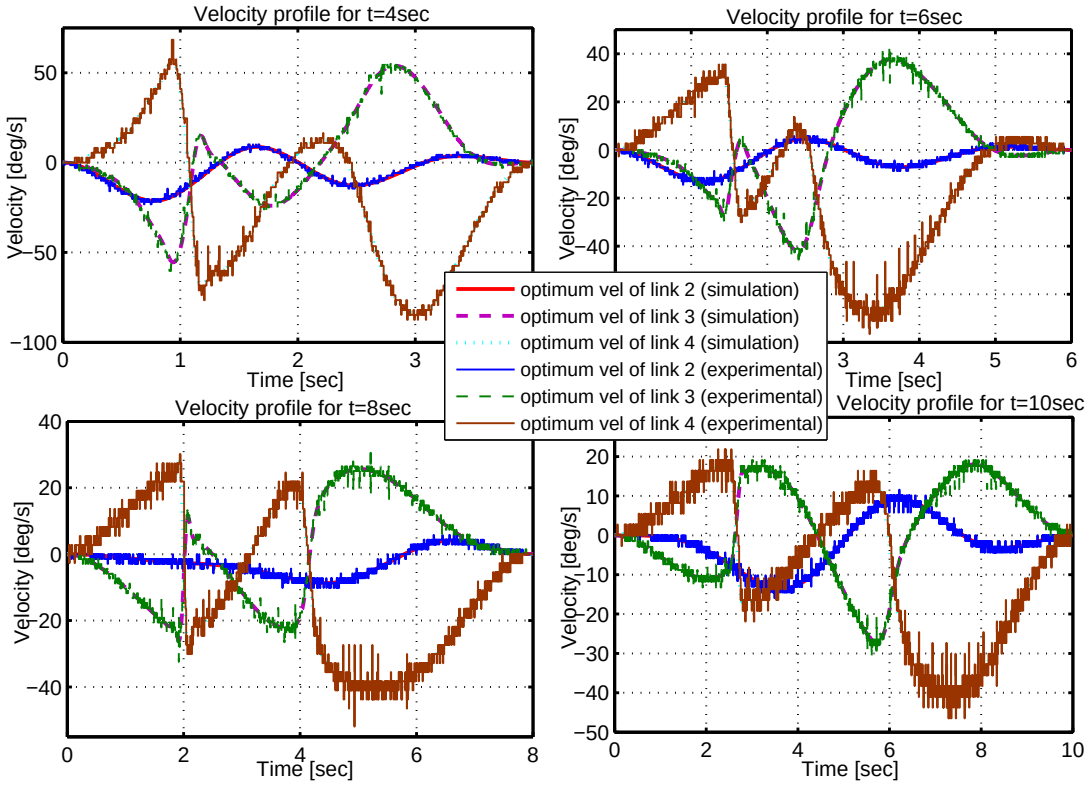

Figure 6-9: *Experimental comparison of speed with varying duration of motion (The plots indicate the optimum velocity output of the theoretical and experimental studies).*

motion. Hence, the torque magnitude of link 2 has almost zero magnitude between these normalised times until the end of the desired motion.

In the duration of motion of 8 second profile, unlike other movements in 4, 6 and 10 seconds, there are nearly two singular configuration points at 2 and 4.15 seconds of the motion. The corresponding manipulator's configurations at these durations of motion are shown in "temporal position for 8 sec profile (a)" as shown in Fig. 6-8. In the first singular point of the duration (at 2 second of motion), unlike other decline velocity curve profiles in 4, 6 and 10 seconds of the motion, it is quite sudden and sharper.

In the duration of motion of 4.15 sec, another singularity point occurs as shown in the output determinant for 8 second motion profile in Fig. 6-7. For this duration of motion, the velocity profiles of link 3 and link 4 are once again approaching zero degrees as shown in the velocity profile for 8 second in Fig. 6-9. In the second singular configuration of the manipulator, all of the torque profiles of the manipulator have been

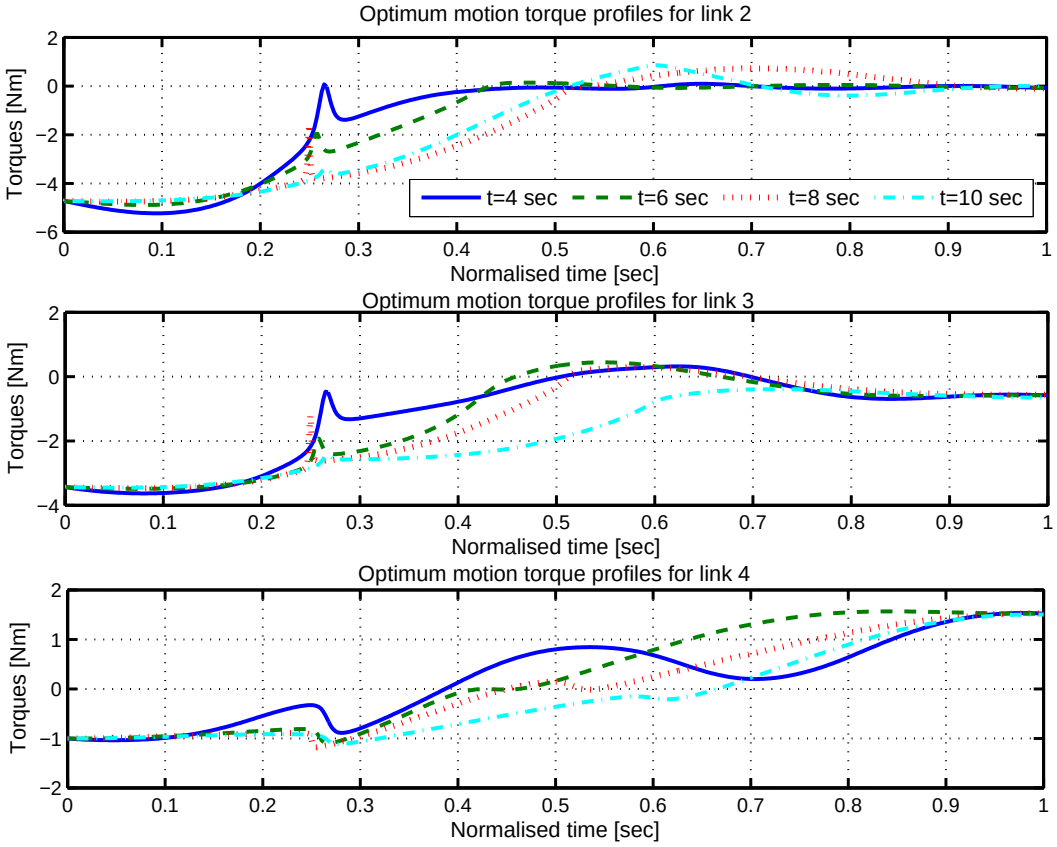

Figure 6-10: *Simulated planned trajectories of optimum path for various duration of motion with torque profiles (normalised time for direct comparisons between trajectories).*

observed as zero torque magnitude for a short period of time as shown in Fig. 6-10 and these values of the torque profiles are consistent with the configuration of the duration of the motion of 4.15 sec in "temporal position for 8 sec profile (a)" as shown in Fig. 6-8.

The duration of the motion of 10 sec profile indicates that the singularity point has occurred at the duration of motion of 2.65 second. Similar to the other movements, the velocity profile of link 3 and link 4 are approaching zero velocity at this duration of motion as shown in velocity profile for 10 second in Fig. 6-9. In addition, the torque profile of link 3 and link 4 have a much smoother and lower torque magnitude amongst the others as shown in Fig. 6-10.

The recorded optimum current profiles of the various durations of motion of the experimental results are shown in Fig. 6-11. In order to calculate the optimum current profile for each duration of motion, the experimental trajectories were carried out 5

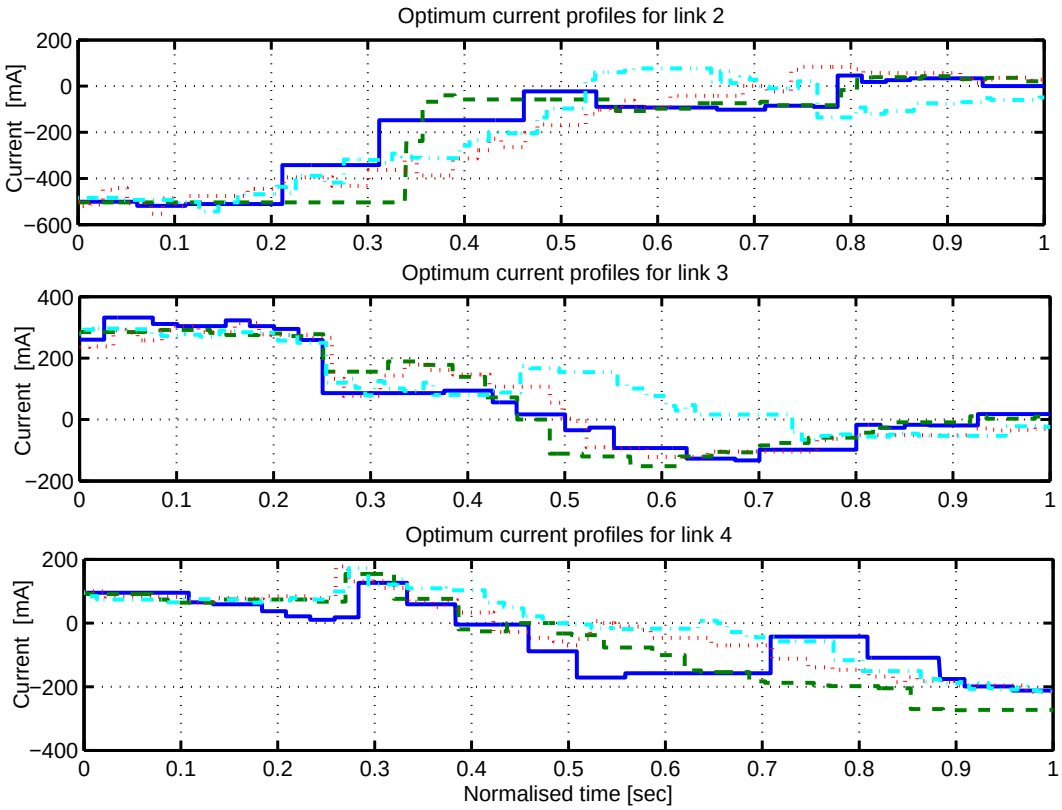

Figure 6-11: *Experimental comparison of optimum current profiles with varying durations of motion (The results are plotted against the normalised time to allow direct comparison to be made for different motion durations).*

times, and the current required in each of the actuators was recorded 5 times and averaged for each sampling of the simulation time. The optimum current profiles of link 2 of the various durations of motion are consistent with the optimum torque profiles of link 2. The current profiles of link 3 and link 4 are also consistent with the theoretical recorded torque profiles as shown in Fig. 6-10. The optimum initial required current profile of link 3 is high and about 300 mA. Furthermore, the optimum initial required current profile of link 4 is also high and starting from 100 mA. If we compare these values with the values of optimum initial torque requirements of link 3 and link 4 in Fig. 6-10, the comparison will clearly show us the experimental results strongly support the theoretical outcomes. The corresponding actuator acceleration and jerk profiles are plotted as shown in Fig. 6-12. The acceleration and jerk profiles of optimum motions have a sudden and a sharp increase between 0.2 and 0.3 normalised times in the link 3

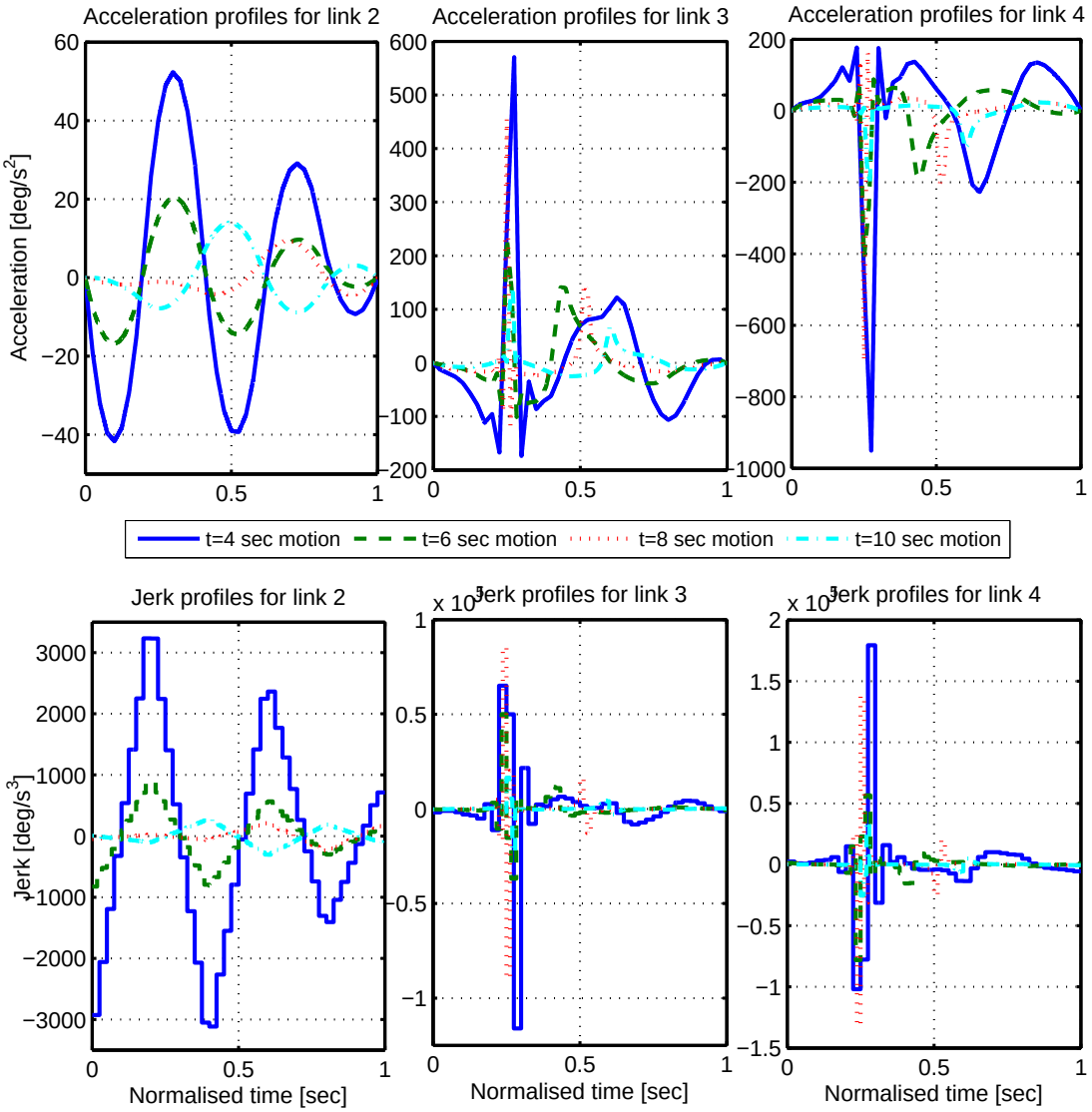

Figure 6-12: *Simulated planned trajectories of optimum path for various duration of motions with acceleration and jerk profiles (normalised time for direct comparison between trajectories).*

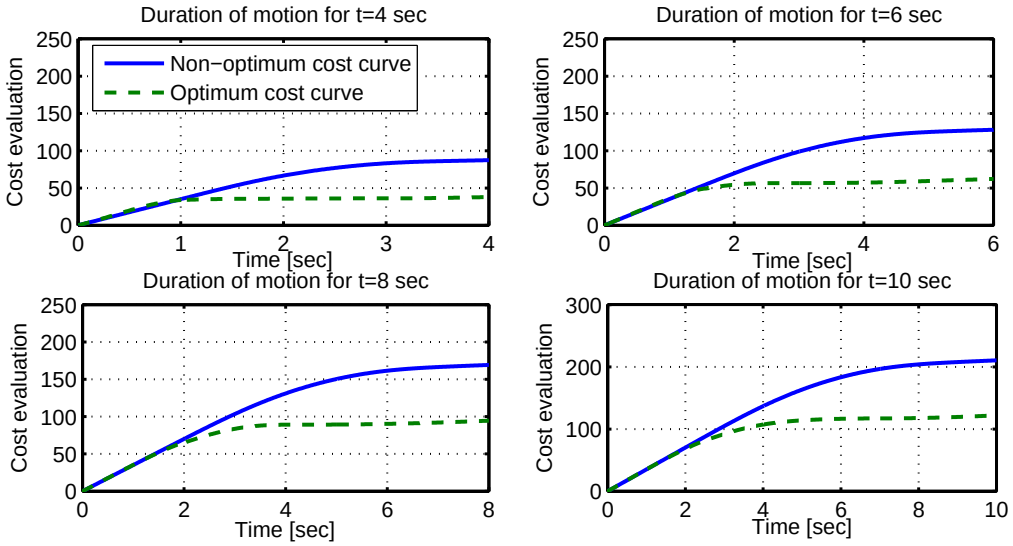

Figure 6-13: *Cost evaluation of theoretical simulation with the different duration of motions.*

and the link 4 of the robotic manipulator. As previously mentioned in section 6.3, this excessive and sudden increase of the acceleration and jerk profile can be explained to be due to increasing the speed of actuator 3 and actuator 4 in the almost first quarter of each motion and then a sudden and sharp decline occurs in the velocity profiles of the link 3 and link 4 due to the singular configurations. The acceleration and jerk profiles of link 3 and link 4 support the theoretical and experimental velocity results in Fig. 6-9. The results of the acceleration and jerk profiles are also consistent with the results of the output determinants as show in Fig. 6-7, the corresponding manipulator positions for the singularity points as shown in Fig. 6-8, and also optimum torque profiles as shown in Fig. 6-10.

The total energy consumed by the robotic manipulator can be accurately calculated by the cost function which is given by Eq. (4.5). Thus, Fig. 6-13 indicates an illustrative instance of the evaluation of the cost functions for the varying duration of motion for the theoretical study. As is previously mentioned, the complexity of the optimization problem as well as the dynamic and kinematic constraints should be handled effectively during the given task. In addition to this, the constrained optimization algorithm is also sensitive to a given initial free optimization parameters, hence, all these conditions may lead the optimization algorithm to end up in different local minima points

during the optimization procedure. At the same time, redundant/hyper-redundant manipulators are under numerous dynamic and kinematic constraint conditions, therefore, depending on a small difference in the starting conditions, they can produce a completely different performance for the outcome of the desired task (on the basis of theoretical experiments).

Although the optimization algorithm produced the result as a feasible optimum solution for a given task, this solution can be considered as a local minima point, and the global minimum point is not clearly met during the simulations. However, as mentioned earlier, this optimum outcome can be taken into consideration as a combination of highly kinematic and dynamic optimum characteristics of the redundant robotic manipulator. As expected, the motion duration has a significant impact on the outcome of the cost function. It can be seen from the Tab. 6.2 and Tab. 6.3 that as the time of the motion is increased the energy consumption is increased.

Time	Non-optimum (Simulation)	Optimum (Simulation)	Energy saving
	N^2m^2s	N^2m^2s	%
Cost(4s)	87	38	%56.57
Cost(6s)	128	62	%51.48
Cost(8s)	169	92	%45.78
Cost(10s)	211	122	%42.21

Table 6.2: *Cost values of the theoretical output of non-optimum and optimum redundant motion with various duration of motion. Theoretical cost calculated from the required actuator torque to be applied at joint i.*

Time	Non-optimum (Katana)	Optimum (Katana)	Energy saving
	amp^2s	amp^2s	%
Cost(4s)	72	31	%43.001
Cost(6s)	99	60	%39.22
Cost(8s)	144	92	%36.21
Cost(10s)	176	119	%32.54

Table 6.3: *Cost values of the experimental output of non-optimum and optimum redundant motion with various duration of motion. Experimental cost calculated provided current data for each axis.*

Time	**Path**	$\mathbf{r_1}$	r_2	r_3	r_4	$\mathbf{r_5}$	$\mathbf{r_6}$	r_7	r_8	r_9	$\mathbf{r_{10}}$
	$x_{non_{opt}}$	**0**	0	0	0	**0.1485**	**0.3015**	0.5000	0.4007	0.5981	-
	$y_{non_{opt}}$	**0**	0	0	0	**-0.0165**	**-0.0335**	-0.0556	-0.0445	-0.0665	-
4sec	$\theta_{red_{non_{opt}}}$	**0**	0	0	0	**-0.1152**	**-0.2339**	-0.3878	-0.3109	-0.4640	**-0.3491**
	$\theta_{red_{opt}}$	**0.4707**	-	-	-	**-0.0402**	**-0.4552**	-	-	-	**-0.3228**
	x_{opt}	**-0.2228**	-	-	-	**0.4302**	**0.3347**	-	-	-	-
	y_{opt}	**-0.1670**	-	-	-	**0.0414**	**0.1623**	-	-	-	-
	$x_{non_{opt}}$	**0**	0	0	0	**0.1485**	**0.3015**	0.5000	0.4007	0.5981	-
	$y_{non_{opt}}$	**0**	0	0	0	**-0.0165**	**-0.0335**	-0.0556	-0.0445	-0.0665	-
6sec	$\theta_{red_{non_{opt}}}$	**0**	0	0	0	**-0.1152**	**-0.2339**	-0.3878	-0.3109	-0.4640	**-0.3491**
	$\theta_{red_{opt}}$	**0.4079**	-	-	-	**-0.0481**	**-0.4151**	-	-	-	**-0.3409**
	x_{opt}	**-0.0881**	-	-	-	**0.4007**	**0.4928**	-	-	-	-
	y_{opt}	**-0.0680**	-	-	-	**0.1794**	**-0.0912**	-	-	-	-
	$x_{non_{opt}}$	**0**	0	0	0	**0.1485**	**0.3015**	0.5000	0.4007	0.5981	-
	$y_{non_{opt}}$	**0**	0	0	0	**-0.0165**	**-0.0335**	-0.0556	-0.0445	-0.0665	-
8sec	$\theta_{red_{non_{opt}}}$	**0**	0	0	0	**-0.1152**	**-0.2339**	-0.3878	-0.3109	-0.4640	**-0.3491**
	$\theta_{red_{opt}}$	**0.0576**	-	-	-	**-0.1426**	**-0.5276**	-	-	-	**-0.3076**
	x_{opt}	**-0.0194**	-	-	-	**0.3308**	**0.4561**	-	-	-	-
	y_{opt}	**-0.0543**	-	-	-	**0.1457**	**0.0803**	-	-	-	-
	$x_{non_{opt}}$	**0**	0	0	0	**0.1485**	**0.3015**	0.5000	0.4007	0.5981	-
	$y_{non_{opt}}$	**0**	0	0	0	**-0.0165**	**-0.0335**	-0.0556	-0.0445	-0.0665	-
10sec	$\theta_{red_{non_{opt}}}$	**0**	0	0	0	**-0.1152**	**-0.2339**	-0.3878	-0.3109	-0.4640	**-0.3491**
	$\theta_{red_{opt}}$	**-0.0154**	-	-	-	**-0.8435**	**-0.0670**	-	-	-	**-0.3428**
	x_{opt}	**-0.0343**	-	-	-	**0.2665**	**0.3304**	-	-	-	-
	y_{opt}	**-0.0668**	-	-	-	**0.0532**	**0.1841**	-	-	-	-

Table 6.4: *Non-optimum and optimum B-spline parameters of redundant trajectories with varying durations of motion.*

After optimization, the cost value is reduced remarkably, and the considerable improvement made was approximately 56.57% in 4 seconds motion. For the theoretical simulations, the cost value calculated from the required actuator torques was to be applied at joints. To calculate the cost value for the experimental study, the terminal currents of each link are measured and recorded. These current values are used to calculate the cost in the same way by using Eq. (4.5). The corresponding experimental cost values for varying durations of motions are demonstrated in Tab. 6.3 and maximum energy reduction was observed to be approximately 43.001%, which corresponds to the duration of motion of 4 second profile in the experimental study.

In addition to this, the B-spline parameters of non-optimum ($x_{non_{opt}}, y_{non_{opt}}$), optimum ($x_{opt}, y_{opt}$) and redundant link angles of non-optimum and optimum motions ($\theta_{red_{non_{opt}}}, \theta_{red_{opt}}$) with various duration of motion are given in Tab. 6.4. The three free control parameters r_1, r_5 and r_6 (printed in bold in Tab. 6.4) are optimized by the optimization algorithm. For the redundant link end, there is another free parameter, r_{10}, to be optimized by the optimization algorithm. This parameter will guarantee the end position of the redundant link.

6.4 Optimal Trajectory Planning for Hyper-Redundant Manipulators on Cartesian Coordinates

This section deals with an optimal trajectory planning algorithm for highly redundant manipulators. A 8-links hyper-redundant (with six DOFs redundancy) manipulator is introduced to verify the proposed virtual link concept as given in section 6.1.1. The technique is based on minimization of a cost function that takes into account the total energy consumed along the whole trajectory. Kinematic and dynamic constraints expressed by means of upper and lower limits on angle, velocity, acceleration and input torque are taken into consideration as shown in Tab. 6.5. The simulation is carried out by the program Dysim, which utilizes the Lagrangian formulation of the equations of motion and is suitable for multi-physics systems.

The optimization routine is based on inverse dynamic analysis which requires the

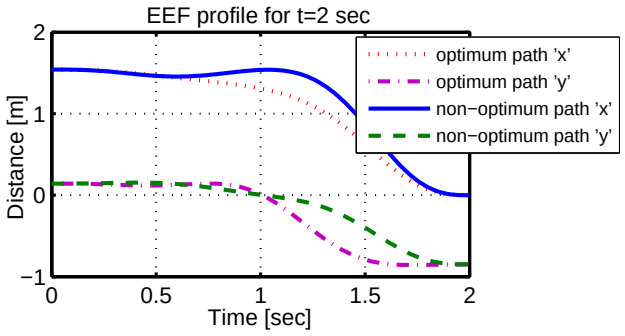

Figure 6-14: *Comparison of simulated non-optimum and optimum position of the hyper-redundant trajectory.*

acceleration profile of the motion as defined by fifth order B-spline functions, which guarantee continuity of velocity and acceleration and provide zero velocity and acceleration for the start and end positions. Based on the features of the optimization problem, a non-linear constrained minimization function was proposed to solve the corresponding optimization model. The manipulator task for this example is to move the load mass from an initial point $\mathbf{P_{initial}}$ $(x_i = 1.54,\ y_i = 0.14)$m to a final point $\mathbf{P_{final}}(x_f = 0,\ y_f = -0.85)$m in Cartesian space coordinate as shown in Fig. 6-15.

The viscous friction effects of the joints are also included with a coefficient of friction '0.6' Ns/m and gears are also taken into account with all gear ratios $R_{1...8} = 50$. The motion duration is specified as T=2 s. Mass centre of gravity of the links are in the middle of each link and the load mass is $m_{load} = 0.3$ kg at the end of the last link. An 8-links hyper-redundant manipulator has eight identical links, and each link length is selected as 0.2 m. For the each link, motor and link inertias were selected as 0.0001 kgm^2 and 0.0025 kgm^2, respectively. All the links have identical mass, $m_1 = \cdots = m_8 = 0.5$ kg and each motor mass was selected as 0.2 kg. The angle, velocity, acceleration and

Constraints	**Links (1,2,3,4,5,6,7,8)**
Relative angles (deg)	+/-122.5°
Velocity (deg/s)	+/-110
Acceleration (deg/s^3)	+/-3500
Torque (Nm)	120

Table 6.5: *Limiting parameters used for Hyper-redundant manipulator.*

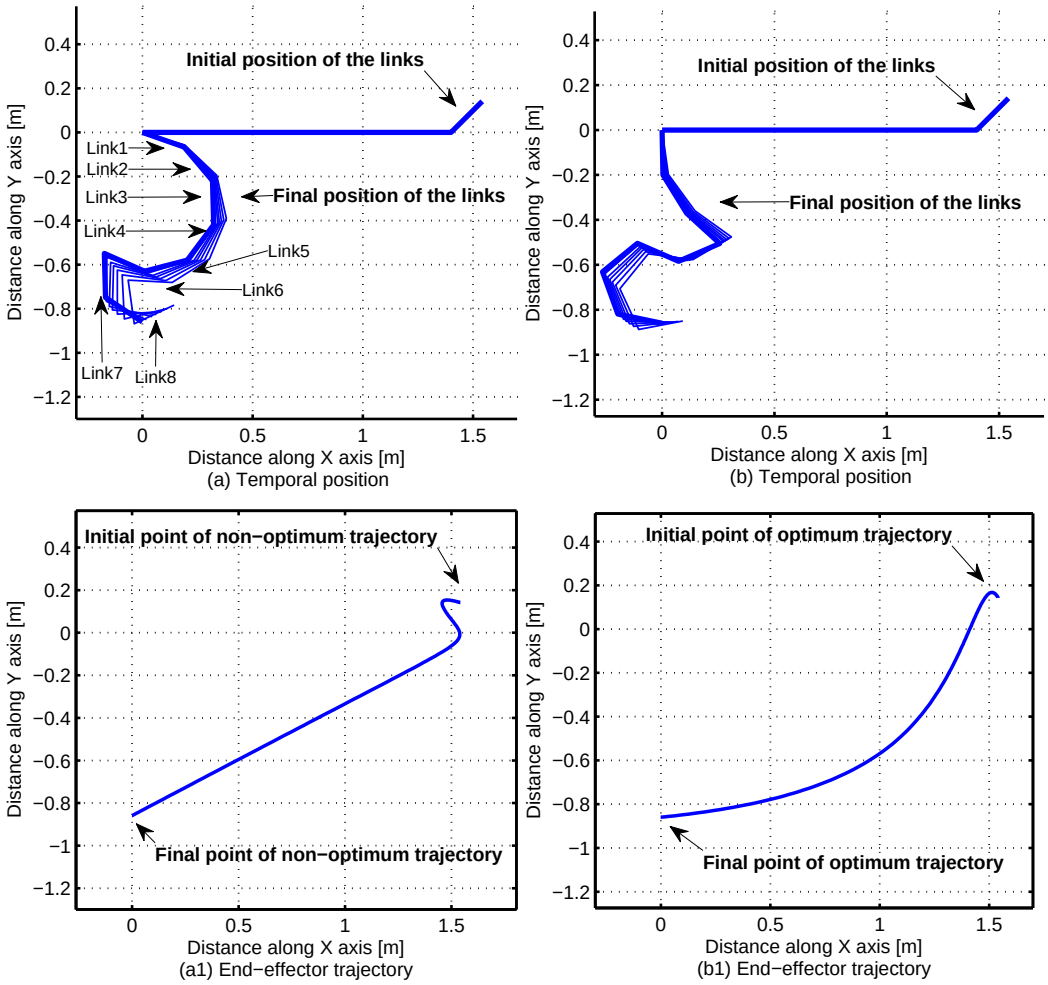

Figure 6-15: *(a),(a1) Motion trajectory corresponding to initial parameter values, (b)(b1), Motion trajectory corresponding to optimum parameters values.*

torque constraints are given in Table 6.5. The Dysim program selects 26 generalised coordinates (three for each links and two for the load) for the robotic manipulator. The system consists of 50 constraints, 58 variables. The Lagrangian function and the dynamic equations of motion including constraint equations and differential-algebraic equations are automatically developed by the DYSIM program.

$$q = [x_1, y_1, \theta_1, x_2, y_2, \theta_2, \ldots x_8, y_8, \theta_8, x_L, y_L] \tag{6.6}$$

The program also calculates the initial conditions of the dependent coordinates based on the user defined initial position conditions of the user selected eight independent

coordinates, angle of θ_1 to angle of θ_8. In this case, angles of links θ_1 to θ_6 and x_L, y_L were selected as the motion defining variable. In this hyper-redundant scheme, θ_1 to θ_6 indicate the relative angle of the redundant links of the robotic manipulator.

To start to the optimization algorithm, a feasible initial path should be chosen in the Cartesian space between the initial and final point of the trajectory. However, for a given workspace position, there are an infinite number of inverse kinematics solutions corresponding to all possible configurations of the hyper-redundant manipulator due to the computational complexity of the hyper-redundancy. Therefore, for the case of hyper-redundant optimization, the initial free optimization parameters are selected randomly as zero for each link and the initial optimization parameters are shown in Tab. 6.7.

The prescribed manipulative task and the obtained optimum path are shown in Fig. 6-14. The corresponding randomly generated feasible non-optimum and optimum manipulative task are shown in temporal trajectory positions and also corresponding EEF's tracking trajectory as presented in Fig. 6-15. Note that each randomly generated feasible end-effector trajectory passed through the specified initial and final points, but it does not track the initial trajectory. As is seen from the Fig. 6-15, the EEF tracking curve of the motion trajectory corresponding to non-optimum parameters has almost a straight line trajectory apart from the beginning of the motion. This movement of the manipulator provides high a torque magnitude and sudden ascension on the non-optimized torque curves for all actuators between the duration of 1.3 and 1.4 seconds in the simulation as shown with dashes in Fig. 6-16.

Excessive growth of the torque values can be shown to be due to the tension between the links for that duration of the motion. The corresponding non-optimum temporal motion between 1.3 and 1.4 seconds and evaluation of the non-optimum cost curve are shown in Fig. 6-17(a) and Fig. 6.6, respectively. As is seen from the Fig. 6.6, the non-optimum cost curve increases rapidly and gives an initial cost value of G=8084. On the other hand, the EEF tracking curve of the motion trajectory corresponding to optimum parameters has a smooth EEF tracking profile as shown in Fig. 6-15(b1) and the corresponding optimum parameters are given in Tab. 6.7. This optimum motion of

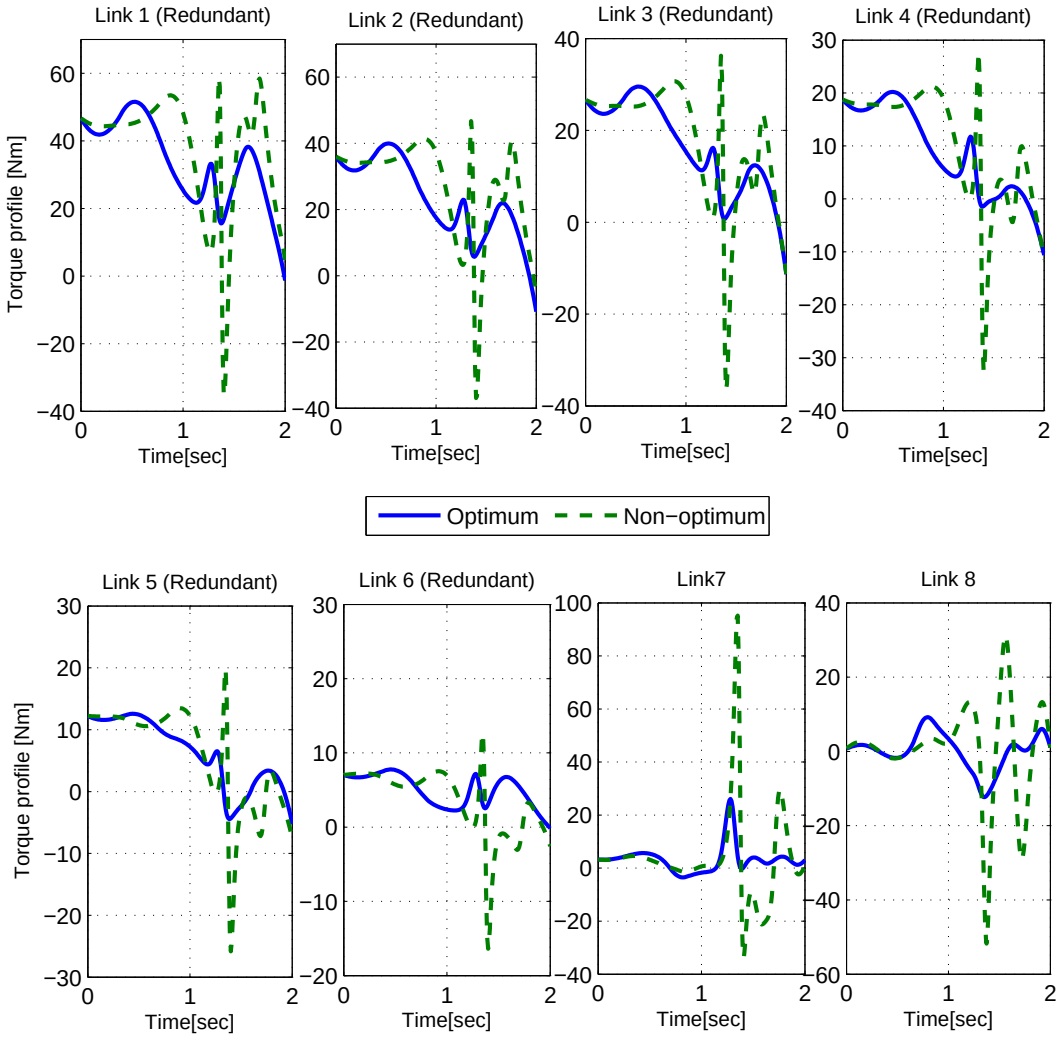

Figure 6-16: *Simulated planned trajectories of non-optimum and optimum torque profiles for hyper-redundant manipulator.*

the manipulator provides admissible torque magnitude and avoids sudden increases in torque profile during the motion as shown in Fig. 6-16. The corresponding optimized cost curve is increasing smoothly as shown in Fig.6.6 and less incline is observed on the curve of the cost value due to the optimum parameters. After optimization, the cost function is reduced to G=5232, which corresponds to 35.3% reduction along the desired trajectory.

The simulation results make clear the effect of the proposed virtual link concept for the redundant/hyper-redundant robotic manipulators. The proposed method not only

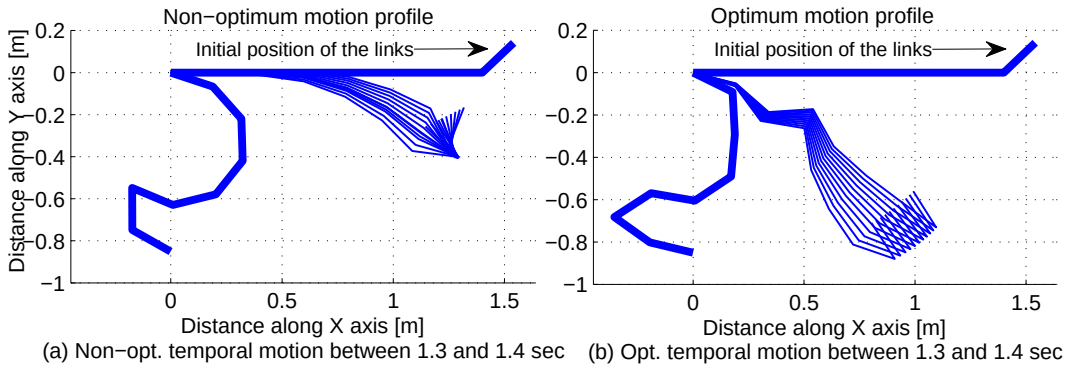

Figure 6-17: *(a) Temporal position corresponding to initial parameter values between 1.3 and 1.4 sec, (b), Temporal position corresponding to optimum parameters values between 1.3 and 1.4 sec.*

Time	Non-optimum(Simulation)	Optimum(Simulation)	Energy saving
	N^2m^2s	N^2m^2s	%
Cost(2 sec)	8084	5232	%35.3

Table 6.6: *Cost values of the theoretical output of non-optimum and optimum Hyper-redundant motion. Theoretical cost calculated from the required actuator torque to be applied at joint i.*

achieves a reduction in the energy consumption for the hyper-redundant manipulators, but also has the ability of handling a large number of DOFs manipulators without any problems. A variety of constraints and different cost functions can easily be added to the proposed optimization procedure.

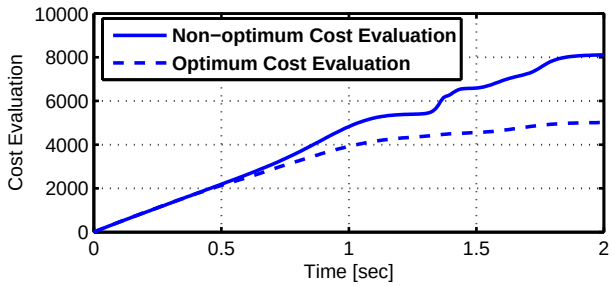

Figure 6-18: *Cost evaluation of theoretical simulation of hyper-redundant manipulators along the desired trajectory.*

Parameters	$\mathbf{link_{1red}}$	$\mathbf{link_{2red}}$
$non-optimum$	0 0 0 0	0 0 0 0
$optimum$	0.1189 , -0.1876, -0.4683, -0.4730	0.0897, 0.1833 ,-1.1200, -1.0459

Parameters	$\mathbf{link_{3red}}$	$\mathbf{link_{4red}}$
$non-optimum$	0 0 0 0	0 0 0 0
$optimum$	0.0517, 0.0395, -0.3639, -0.1359	0.2150, 0.0778, -0.6381, -0.8757

Parameters	$\mathbf{link_{5red}}$	$\mathbf{link_{6red}}$
$non-optimum$	0 0 0 0	0 0 0 0
$optimum$	0.1200, -0.0216 , 1.5747 , -0.7866	0.0406, -0.1623 , -1.8118, 0.7745

Parameters	$\mathbf{link_{7}}$	$\mathbf{link_{8}}$
$non-optimum$	0 0 0 0	0 0 0 0
$optimum$	-0.3364, -0.4189 , 1.0744	-0.1321 , 0.2476 , -0.2123

Table 6.7: *Non-optimum and optimum B-spline parameters of Hyper-redundant trajectory.*

6.5 Error Analysis of Optimum Redundant Trajectories with Varying Durations of Motion

Trajectory optimization inevitably results in the end-effector position error to some extent. In this section, the error results are discussed for the 3-link redundant manipulator scheme on the second, third and fourth link of the Katana 450 robotic manipulator. Between the initial and final point of the trajectory, the optimum motion was implemented in Cartesian coordinates with various durations of motion on the Katana manipulator.

The robot trajectory is composed of cubic polynomials, and for every one second of the motion, 10 cubic polynomials are used to feed the Katana actuators.

Figure 6-19 shows the comparison of the EEF path deviation along the desired trajectory with various demanded path velocities with respect to the normalised time for each trajectory. Normalised time was used for direct comparisons between trajectories. The figure consists of two different types of error. Type 1 error demonstrates the overall tracking error in Cartesian space between the desired trajectory and actual trajectory output from the experimental test rig. In addition, type 2 error is due to the difference between fifth-order B-spline trajectory and cubic polynomial conversion in Cartesian

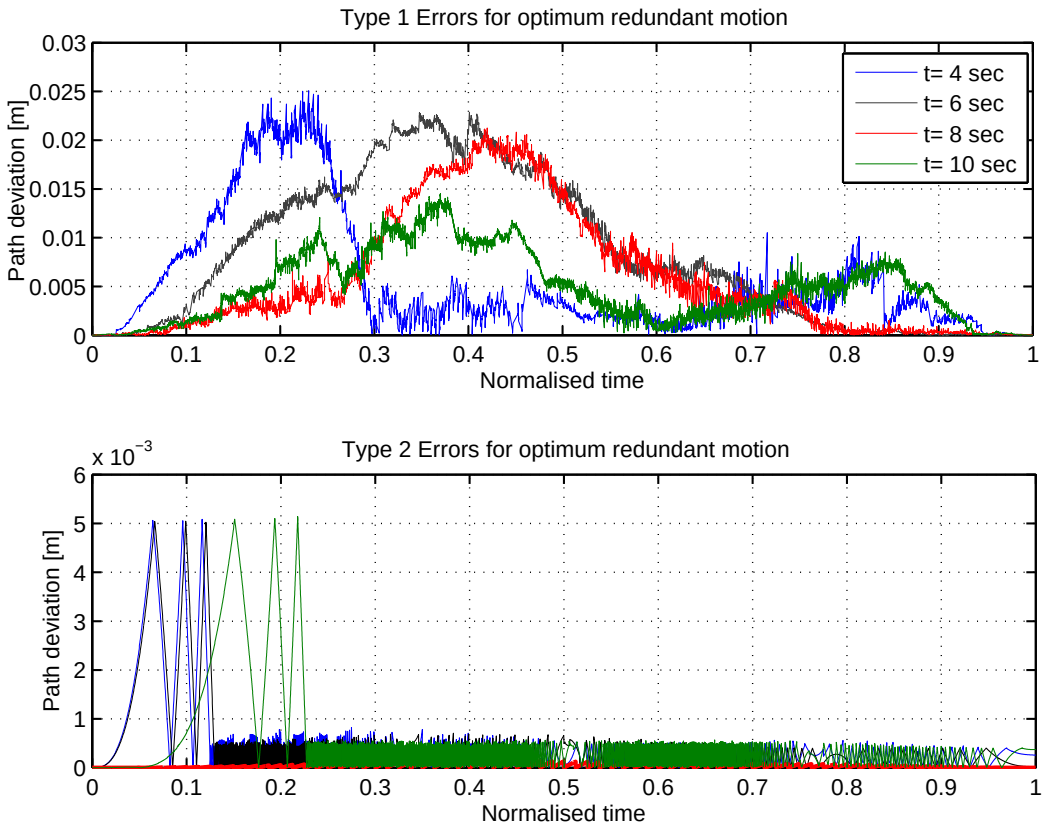

Figure 6-19: *(a) Type 1 Errors: Overall experimental tracking errors in Cartesian space with various duration of motion. (b) Type 2 Errors: Theoretical absolute error comparison for the simulated optimum motion trajectory (B-spline to cubic conversion) with various duration of motion.*

coordinates.

Comparing the plots in Fig. 6-19(a), we can see that the larger path deviation is found in 4 secs of duration of motion. This indicates that the position error depends on the speed of the robotic manipulator, i.e., higher speeds result in higher path deviation and vice-versa. For the type 1 error, the maximum error was found as 0.0251 m for the four second motion and followed by 0.0229 m, 0.0212 m, 0.0145 m for t=6, t=8, t=10 seconds of duration of slower motion respectively, as shown in Tab. 6.8.

It is seen from the Fig. 6-19(b) that all of the plots have a similar behaviour during the transformation from the B-spline to cubic polynomials. In type 2 error, 4, 6 and 10 seconds of motion have almost the same maximum error during the motion. However, 8 second motion has the smallest value in path deviation for the maximum error of

Type of Errors	4 sec	6 sec	8 sec	10 sec
Maximum error of type 1(m)	0.0251	0.0229	0.0212	0.0145
Maximum error of type 2(m)	0.0051	0.0050	0.000151	0.0051
Mean error of type 1(m)	0.005386	0.008261	0.005802	0.004812
Mean error of type 2(m)	0.000471	0.0004607	3.71944e-05	0.000526911

Table 6.8: *Error summary for optimum redundant trajectory motion with various duration of motion.*

type 2 and also mean error of type 2.

6.6 Concluding Remarks

This chapter investigated the optimal trajectory planning for redundant/hyper-redundant robotic manipulators using the proposed alternative cost function as defined in Ch. 4. In this alternative cost function, the constraints are handled within the cost function to improve computational efficiency by preventing calls to the inverse dynamic model when constraints are not satisfied.

Firstly, the optimization problem of redundant manipulators has been presented and formulated by using an inverse dynamic analysis for a redundant manipulator system. The DYSIM software is utilised to calculate the equations of motion automatically. A uniform fifth-order B-spline function was used to define the end-effector trajectory and also relative angles of each redundant link. This was followed by converting B-spline trajectories to cubic polynomials to meet the Katana 450 input requirements.

A virtual link concept has been introduced to replace all redundant links to simplify the process of eliminating physically unrealizable paths before calling the inverse dynamic simulation for the redundant/hyper redundant serial link robotic manipulators. This virtual link concept leads to controlling a massive number of DOFs robotic manipulator while reducing the energy consumption in the desired trajectories.

The proposed scheme is verified with two different computer simulations on 3-links redundant and 8-links hyper-redundant manipulators. In addition to this, the method was also applied on 3-link Katana robotic manipulator based on link 2, link 3 and link

4 to verify the proposed method. The optimization results show that, a significant improvement can be achieved in the cost function for the simulation and experimental study and also a large number of DOFs manipulator can easily be handled for any given task.

CHAPTER 7

CONCLUSIONS

The work presented in this thesis provides new techniques regarding trajectory optimization for redundant/hyper-redundant manipulators. Firstly, a proposed optimization method demonstrates the effective alternative constraint handling method for non-redundant, redundant/hyper-redundant manipulators and it is applicable to any type of robotic manipulator. This proposed optimization method prevents running the inverse dynamic simulation to crash or terminate prematurely as the required motion cannot be physically achieved. Secondly, a virtual link concept has been introduced to replace all redundant links for redundant/hyper-redundant manipulators to simplify the process of eliminating physically unrealizable paths before calling the inverse dynamic simulation.

Compared with the other trajectory optimization techniques, the proposed optimization method provides important contributions to the literature. It is computationally efficient as kinematic and dynamic constraints are included in the cost function to prevent the running inverse dynamic model when the parameters produce a motion that does not satisfy all of the constraints. The proposed algorithm includes efficient constraint handling within the cost function. The proposed method was first carried out on a simple 2-DOF planar robotic manipulator with revolute joints. Although the optimization results are similar for proposed and conventional methods, it was observed

that the optimization algorithm called the inverse dynamic unnecessarily (84 out of 246 iteration) in the conventional method. Unnecessary solving of the inverse dynamics has a significant effect on the computational efficiency of the optimization. The computational cost in the proposed algorithm is reduced significantly due to not running the inverse model in the alternative cost function calculations when the constraints are not satisfied.

In order to show how the proposed alternative constraint handling method works, an inverse dynamic model of a two degrees of freedom Katana 450 manipulator based on link 2 and link 4 is utilised in various trajectory profiles (such as straight-line on end-effector motion, straight-line on joint motion, optimal motion) with different settings (various durations of motion) for a non-redundant robotic manipulator. The proposed method was initially implemented utilising computer simulations, with the implementation of the proposed method to the Katana manipulator being simulated.

Although the cost values are increased by increasing the duration of motion, theoretical and experimental optimization results showed a significant reduction of the energy consumption for each implemented trajectory. After optimising the desired task for the theoretical non-redundant study, the energy saving percentages were increased, 10.2 %, 32 %, 48.1 %, 55.8 % for 2, 4, 6, and 8 second motion durations, respectively. These results were compared to the experimental results utilising the same trajectory motions and execution times. The corresponding experimental energy saving rates are 8.71 %, 23.87 %, 34.50 %, 43.37 % for 2, 4, 6, and 8 second motion durations, respectively.

For the next stage of the work, a three-link redundant model of the Katana 450 robotic manipulator based on links 2 (redundant), 3 and 4 was designed and constructed for the implementation of the redundant case. The novelty of this redundant control scheme is the introduction of the virtual link concept, where all of the redundant links are acting as a single link. In this case of hyper-redundant manipulators, all redundant links are replaced by a virtual link to eliminate physically impossible link configurations before running the inverse dynamic model, preventing the inverse dynamic simulation failure. In addition to this, the proposed optimization method is also utilised in the redundant case to handle the various constraints within the cost function to prevent

running the inverse dynamic when all the constraints are not satisfied.

In order to control the redundant links, a uniform fifth-order B-spline function was also utilised. As is given in the non-redundant case, three control points were utilised to satisfy the initial conditions (position, its first and second derivatives) and another three control points were used to satisfy the end conditions (position, its first and second derivatives). The remaining three free control points are optimized by the trajectory optimization algorithm. In the redundant case, six free parameters for the trajectory of the end-effector in Cartesian coordinates, and four free parameters for the relative angle of each redundant links are utilised by the optimization algorithm. The end position of the redundant links is also optimized, as it is not known in advance.

In order to demonstrate the effectiveness of the virtual link concept, firstly, a three-link redundant model of the Katana 450 robotic manipulator was performed with different settings (various durations of motion). The virtual link was initially performed utilising computer simulations, followed by experimental implementations. After optimising the desired task for the theoretical redundant study, the energy saving percentages are 56.57 %, 51.48 %, 45.78 %, 42.21 % for 4, 6, 8, and 10 second motion durations, respectively. These results were compared to the experimental results utilising the same trajectory motions and execution times. The corresponding experimental energy saving rates are 43.001 %, 39.22 %, 36.21 %, 32.54 % for 4, 6, 8, and 10 second motion durations, respectively.

In order to verify the effectiveness of the proposed virtual link concept, a numerical inverse dynamic model for 8-link (6 links are redundant) hyper-redundant manipulator was also executed. A uniform fifth-order B-spline was also utilised to define the trajectory. Although the hyper-redundant system consists of 50 constraints and 58 variables, the results indicate that a significant reduction can be achieved in the cost function in simulation and large number of redundant DOFs as well as kinematic and dynamic constraints can easily be handled by the proposed alternative constraint handling technique. After optimization, the reduction in the cost function is 35.3 % along the desired trajectory.

Because of the importance of the optimum trajectory planning issue, apart from

the achievements mentioned in this thesis, there are still a large amount of room for further research which can be the extension of this work. In this way, the quality and the functionality of the proposed method can be improved. For example;

- This work can be performed in three dimensions which could be more problematic and challenging for trajectory planning. And also, the orientation of the gripper of the manipulator can be taken into account to execute some tasks.

- The existing trajectory optimization algorithm can be rebuilt by utilising genetic algorithms. Knowing that, genetic algorithms are powerful in finding the global minimum point for a given trajectory. In this way, it will be an opportunity to check their efficiency in optimum trajectory planning for industrial robots.

- Multi-cost criteria (such as energy+time, energy+jerk, energy+time+jerk) can be introduced in the objective function to improve the present work for the trajectory optimization method. For example, in some situations such as maximizing speed of operation whilst at the same time minimizing the error and energy consumption for a given task, the quality of the motion and the productivity may improve by utilising the multi-cost criteria.

- Real time trajectory planning can be considered with the obstacles avoidance scheme. In this scheme, the use of a camera or sensors could be integrated into the system further in order to avoid obstacles effectively.

REFERENCES

[1] Vincent Yap Kah Jiun. *Motion Planning and Control of Katana 450 Robot.* Final year project, Department of Mechanical Engineering, University of Bath, 2010.

[2] *Katana user manual and technical description.* AG, Neuronics, 2009.

[3] Robert M. C. Rayner. *The optimisation of a high speed servomechanism.* Ph.D., University of Bath, 2010.

[4] B. N. Saeed and F. Sun. Introduction to robotics analysis, systems, applications. *Edit. Prentice Hall*, 2001.

[5] J. J. Craig. Introduction to robotics: mechanics and control. 2004.

[6] Z. Zoller and P. Zentay. Constant kinetic energy robot trajectory planning. *Periodica Polytechnica, Mechanical Engineering*, 43(2):213–228, 1999.

[7] S. Niku. *Introduction to Robotics.* Wiley, 2010.

[8] M. N. Sahinkaya. Inverse dynamic analysis of multiphysics systems. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 218(1):13–26, 2004.

[9] K. K. Ayten, P. Iravani, and M. N. Sahinkaya. Optimum trajectory planning for industrial robots through inverse dynamics. In *Proceedings of the 8th Interna-*

tional Conference on Informatics in Control, Automation and Robotics (ICINCO 2011), 2011.

[10] B. Dasgupta, A. Gupta, and E. Singla. A variational approach to path planning for hyper-redundant manipulators. *Robotics and Autonomous Systems*, 57(2):194–201, 2009.

[11] M. E. Kahn and B. Roth. The near-minimum-time control of open-loop articulated kinematic chains. *Journal of Dynamic Systems, Measurement, and Control*, 93:164, 1971.

[12] M. Niv and D. M. Auslander. Optimal control of a robot with obstacles. In *American Control Conference, 1984*, pages 280–287, 1984.

[13] S. Dubowsky and Z. Shiller. Optimal dynamic trajectories for robotic manipulators. *Theory and Practice of Robots and Manipulators*, pages 133–143, 1984.

[14] J. E. Bobrow, S. Dubowsky, and J. S. Gibson. Time-optimal control of robotic manipulators along specified paths. *The International Journal of Robotics Research*, 4(3):3–17, 1985.

[15] J. E. Bobrow. *Optimal control of robotic manipulators*. PhD thesis, University of California, Los Angeles, 1982.

[16] J. E. Bobrow, S. Dubowsky, and J. S. Gibson. On the optimal control of robotic manipulators with actuator constraints. In *American Control Conference, 1983*, pages 782–787, 1983.

[17] K. Shin and N. McKay. Minimum-time control of robotic manipulators with geometric path constraints. *Automatic Control, IEEE Transactions on*, 30(6):531–541, 1985.

[18] F. Pfeiffer and R. Johanni. A concept for manipulator trajectory planning. *Robotics and Automation, IEEE Journal of*, 3(2):115–123, 1987.

[19] J. J. E. Slotine and H. S. Yang. Improving the efficiency of time-optimal path-following algorithms. *Robotics and Automation, IEEE Transactions on*, 5(1):118–124, 1989.

[20] Z. Shiller and H. H. Lu. Robust computation of path constrained time optimal motions. In *Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on*, pages 144–149, 1990.

[21] L. Zlajpah. On time optimal path control of manipulators with bounded joint velocities and torques. In *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, volume 2, pages 1572–1577, 1996.

[22] M. Leahy and G. Saridis. Compensation of unmodeled PUMA manipulator dynamics. In *Robotics and Automation. Proceedings. 1987 IEEE International Conference on*, volume 4, pages 151–156, 1987.

[23] Z. Shiller, H. Chang, and V. Wong. The practical implementation of time-optimal control for robotic manipulators. *Robotics and computer-integrated manufacturing*, 12(1):29–39, 1996.

[24] D. Constantinescu. *Smooth time optimal trajectory planning for industrial manipulators*. PhD thesis, University of British Columbia, 1998.

[25] D. Costantinescu and E. A. Croft. Smooth and time-optimal trajectory planning for industrial manipulators along specified paths. *Journal of Robotic Systems*, 17(5):233–249, 2000.

[26] C. G. L. Bianco and A. Piazzi. Minimum-time trajectory planning of mechanical manipulators under dynamic constraints. *International Journal of Control*, 75(13):967–980, 2002.

[27] C. G. L. Bianco and A. Piazzi. A genetic/interval approach to optimal trajectory planning of industrial robots under torque constraints. In *Proceedings of the European Control Conference, Karlsruhe, Germany*, volume 31, 1999.

[28] C. G. L. Bianco and A. Piazzi. A hybrid algorithm for infinitely constrained optimization. *International Journal of Systems Science*, 32(1):91–102, 2001.

[29] F. Valero, V. Mata, and A. Besa. Trajectory planning in workspaces with obstacles taking into account the dynamic robot behaviour. *Mechanism and machine theory*, 41(5):525–536, 2006.

[30] F. Rubio, F. Valero, J. Sunyer, and J. Cuadrado. Optimal time trajectories for industrial robots with torque, power, jerk and energy consumed constraints. *Industrial Robot: An International Journal*, 39(1):92–100, 2012.

[31] D. E. Goldberg. *Genetic algorithms in search, optimization, and machine learning*. Addison-wesley, 1989.

[32] M. Gen and R. Cheng. *Genetic algorithms and engineering optimization*, volume 7. Wiley-interscience, 2000.

[33] T. Chettibi, H. E. Lehtihet, M. Haddad, and S. Hanchi. Minimum cost trajectory planning for industrial robots. *European Journal of Mechanics-A/Solids*, 23(4):703–715, 2004.

[34] P. Tangpattanakul and P. Artrit. Minimum-time trajectory of robot manipulator using harmony search algorithm. In *Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, 2009. ECTI-CON 2009. 6th International Conference on*, volume 1, pages 354–357, 2009.

[35] R. Saravanan, S. Ramabalan, and C. Balamurugan. Evolutionary collision-free optimal trajectory planning for intelligent robots. *The International Journal of Advanced Manufacturing Technology*, 36(11):1234–1251, 2008.

[36] K. J. Kyriakopoulos and G. N. Saridis. Minimum jerk path generation. In *Robotics and Automation, 1988. Proceedings., 1988 IEEE International Conference on*, pages 364–369, 1988.

[37] K. J. Kyriakopoulos and G. N. Saridis. Minimum jerk for trajectory planning and control. *ROBOTICA-CAMBRIDGE-*, 12:109–109, 1994.

[38] D. Simon and C. Isik. A trigonometric trajectory generator for robotic arms. *International Journal of Control*, 57(3):505–517, 1993.

[39] A. Piazzi and A. Visioli. An interval algorithm for minimum-jerk trajectory planning of robot manipulators. In *Decision and Control, 1997., Proceedings of the 36th IEEE Conference on*, volume 2, pages 1924–1927, 1997.

[40] A. Piazzi and A. Visioli. Global minimum-jerk trajectory planning of robot manipulators. *Industrial Electronics, IEEE Transactions on*, 47(1):140–149, 2000.

[41] Y. Stepanenko. Dissipative properties and optimal control of manipulators. In *Proc. of the Second Symposium on the Control of Artificial Limbs*, 1970.

[42] M. Vukobratovic and M. Kircanski. A method for optimal synthesis of manipulation robot trajectories. *Journal of Dynamic Systems, Measurement, and Control*, 104:188, 1982.

[43] K. Shin and N. McKay. A dynamic programming approach to trajectory planning of robotic manipulators. *Automatic Control, IEEE Transactions on*, 31(6):491–500, 1986.

[44] O. Von Stryk and M. Schlemmer. Optimal control of the industrial robot manutec r3. *Computational optimal control, International series of Numerical Mathematics*, 115:367–382, 1994.

[45] O. von Stryk and R. Bulirsch. Direct and indirect methods for trajectory optimization. *Annals of Operations Research*, 37(1):357–373, 1992.

[46] G. Field and Y. Stepanenko. Iterative dynamic programming: an approach to minimum energy trajectory planning for robotic manipulators. In *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, volume 3, pages 2755–2760, 1996.

[47] C. De Boor. *A practical guide to splines*, volume 27. Springer Verlag, 2001.

[48] B. J. Martin and J. E. Bobrow. Minimum effort motions for open chain manipulators with task-dependent end-effector constraints. In *Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on*, volume 3, pages 2044–2049, 1997.

[49] D. C. Kar, K. Kurien Issac, and K. Jayarajan. Minimum energy force distribution for a walking robot. *Journal of Robotic Systems*, 18(2):47–54, 2001.

[50] B. S. Lin and S. M. Song. Dynamic modelling, stability, and energy efficiency of a quadrupedal walking machine. *Journal of Robotic Systems*, 18(11):657–670, 2001.

[51] S. F. P. Saramago and M. Ceccarelli. An optimum robot path planning with payload constraints. *Robotica*, 20(04):395–404, 2002.

[52] D. P. Garg and M. Kumar. Optimization techniques applied to multiple manipulators for path planning and torque minimization. *Engineering applications of artificial intelligence*, 15(3):241–252, 2002.

[53] A. Biswas, B. L. Deekshatulu, and S. S. Roy. Energy optimal trajectory planning of a robotic manipulator using genetic algorithm. In *AIP Conference Proceedings*, volume 1298, page 492, 2010.

[54] T. Chettibi, P. Lemoine, et al. Generation of point to point trajectories for robotic manipulators under electro-mechanical constraints. *International Review of Mechanical Engineering (I.R.E.M.E.)*, 1:131–143, 2007.

[55] R. Saravanan, S. Ramabalan, and C. Balamurugan. Evolutionary multi-criteria trajectory modelling of industrial robots in the presence of obstacles. *Engineering Applications of Artificial Intelligence*, 22(2):329–342, March 2009.

[56] R. Saravanan, S. Ramabalan, C. Balamurugan, and A. Subash. Evolutionary trajectory planning for an industrial robot. *International Journal of Automation and Computing*, 7(2):190–198, 2010.

[57] A. Gasparetto and V. Zanotto. Optimal trajectory planning for industrial robots. *Advances in Engineering Software*, 41(4):548–556, 2010.

[58] A. Gasparetto and V. Zanotto. A new method for smooth trajectory planning of robot manipulators. *Mechanism and Machine Theory*, 42(4):455–471, 2007.

[59] A. Gasparetto and V. Zanotto. A technique for time-jerk optimal planning of robot trajectories. *Robotics and Computer-Integrated Manufacturing*, 24(3):415–426, 2008.

[60] A. Gasparetto, A. Lanzutti, R. Vidoni, and V. Zanotto. Experimental validation and comparative analysis of optimal time-jerk algorithms for trajectory planning. *Robotics and Computer-Integrated Manufacturing*, 2011.

[61] Z. Shiller. Time-energy optimal control of articulated systems with geometric path constraints. In *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*, pages 2680–2685, 1994.

[62] S. F. P. Saramago and V. Steffen. Optimization of the trajectory planning of robot manipulators taking into account the dynamics of the system. *Mechanism and machine theory*, 33(7):883–894, 1998.

[63] S. F. P. Saramago and V. S. Junior. Optimal trajectory planning of robot manipulators in the presence of moving obstacles. *Mechanism and Machine Theory*, 35(8):1079–1094, 2000.

[64] S. Ramabalan, R. Saravanan, and C. Balamurugan. Multi-objective dynamic optimal trajectory planning of robot manipulators in the presence of obstacles. *The International Journal of Advanced Manufacturing Technology*, 41(5):580–594, 2009.

[65] H. Xu, J. Zhuang, S. Wang, and Z. Zhu. Global time-energy optimal planning of robot trajectories. In *Mechatronics and Automation, 2009. ICMA 2009. International Conference on*, pages 4034–4039, 2009.

[66] R. Saravanan, S. Ramabalan, and C. Balamurugan. Evolutionary collision-free optimal trajectory planning for intelligent robots. *The International Journal of Advanced Manufacturing Technology*, 36(11):1234–1251, 2008.

[67] R. Saravanan and S. Ramabalan. Evolutionary minimum cost trajectory planning for industrial robots. *Journal of Intelligent & Robotic Systems*, 52(1):45–77, 2008.

[68] R. Saravanan, S. Ramabalan, and C. Balamurugan. Evolutionary optimal trajectory planning for industrial robot with payload constraints. *The International Journal of Advanced Manufacturing Technology*, 38(11):1213–1226, 2008.

[69] Y. Nakamura, H. Hanafusa, and T. Yoshikawa. Task-priority based redundancy control of robot manipulators. *The International Journal of Robotics Research*, 6(2):3–15, 1987.

[70] E. S. Conkur and R. Buckingham. Manoeuvring highly redundant manipulators. *Robotica*, 15(4):435–447, 1997.

[71] M. Z. Ding, C. J. Ong, and A. N. Poo. Resolution of redundant manipulators via distance optimization. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 214(8):1037–1047, 2000.

[72] G. Ping, B. Wei, X. Li, and X. Luo. Real time obstacle avoidance for redundant robot. In *Mechatronics and Automation, 2009. ICMA 2009. International Conference on*, pages 223–228, 2009.

[73] T. Yoshikawa. Analysis and control of robot manipulators with redundancy. In *Robotics Research: The First International Symposium*, pages 735–747, 1984.

[74] A. Liegeois. Automatic supervisory control of the configuration and behavior of multibody mechanisms. *IEEE Trans Sys Man and Cybernetics*, (12):868–871, 1977.

[75] J. Hollerbach and K. Suh. Redundancy resolution of manipulators through torque optimization. *Robotics and Automation, IEEE Journal of*, 3(4):308–316, 1987.

[76] O. S. Yahsi and K. Ozgoren. Minimal joint motion optimization of manipulators with extra degrees of freedom. *Mechanism and machine theory*, 19(3):325–330, 1984.

[77] R. G. Roberts and A. A. Maciejewski. A local measure of fault tolerance for kinematically redundant manipulators. *Robotics and Automation, IEEE Transactions on*, 12(4):543–552, 1996.

[78] J. D. English and A. A. Maciejewski. Fault tolerance for kinematically redundant manipulators: Anticipating free-swinging joint failures. *Robotics and Automation, IEEE Transactions on*, 14(4):566–575, 1998.

[79] C. L. Lewis and A. A. Maciejewski. Dexterity optimization of kinematically redundant manipulators in the presence of joint failures. *Computers & electrical engineering*, 20(3):273–288, 1994.

[80] A. A. Maciejewski. Fault tolerant properties of kinematically redundant manipulators. In *Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on*, pages 638–642, 1990.

[81] Y. Yan, T. Yamamoto, T. Murakami, and K. Ohnishi. Autonomous decentralized approach in the PTP control of redundant manipulators. In *Advanced Motion Control, 1998. AMC'98-Coimbra., 1998 5th International Workshop on*, pages 153–158, 1998.

[82] Erdinc Sahin Conkur. *Real time path planning and obstacle avoidance algorithms for redundant manipulators.* Ph.D., University of Bristol, 1997.

[83] V. D. Tourassis, M. H. Ang, et al. Identification and analysis of robot manipulator singularities. *The International journal of robotics research*, 11(3):248–259, 1992.

[84] K. Kosuge and K. Furuta. Kinematic and dynamic analysis of robot arm. In *Robotics and Automation. Proceedings. 1985 IEEE International Conference on*, volume 2, pages 1039–1044, 1985.

[85] M. Uchiyama, K. Shimizu, and K. Hakomori. Performance evaluation of manipulators using the jacobian and its application to trajectory planning. *Robotics Research*, 2:447–454, 1985.

[86] T. Yoshikawa. Manipulability and redundancy control of robotic mechanisms. In *Robotics and Automation. Proceedings. 1985 IEEE International Conference on*, volume 2, pages 1004–1009, 1985.

[87] R. Mayorga and A. Wong. A singularities avoidance method for the trajectory planning of redundant and nonredundant robot manipulators. In *Robotics and Automation. Proceedings. 1987 IEEE International Conference on*, volume 4, pages 1707–1712, 1987.

[88] R. V. Mayorga and A. K. C. Wong. A singularities avoidance approach for the optimal local path generation of redundant manipulators. In *Robotics and Automation, 1988. Proceedings., 1988 IEEE International Conference on*, pages 49–54, 1988.

[89] M. W. Spong and M. Vidyasagar. *Robot dynamics and control*. Wiley-India, 2008.

[90] P. Mckerrow. *Introduction to robotics*. Addison-Wesley Longman Publishing Co., Inc., 1991.

[91] N. S. Bedrossian. Classification of singular configurations for redundant manipulators. In *Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on*, pages 818–823, 1990.

[92] Y. Nakamura and H. Hanafusa. Inverse kinematic solutions with singularity robustness for robot manipulator control. *Journal of Dynamic Systems, Measurement, and Control*, 108:163, 1986.

[93] C. W. Wampler. Manipulator inverse kinematic solutions based on vector formulations and damped least-squares methods. *Systems, Man and Cybernetics, IEEE Transactions on*, 16(1):93–101, 1986.

[94] A. A. Maciejewski and C. A. Klein. Numerical filtering for the operation of robotic manipulators through kinematically singular configurations. *Journal of Robotic Systems*, 5(6):527–552, 1988.

[95] C. Y. Chung, B. H. Lee, M. S. Kim, and C. W. Lee. Torque optimizing control with singularity-robustness for kinematically redundant robots. *Journal of Intelligent & Robotic Systems*, 28(3):231–258, 2000.

[96] J. Wunderlich and C. Boncelet. Local optimization of redundant manipulator kinematics within constrained workspaces. In *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, volume 1, pages 127–132, 1996.

[97] F. B. M. Duarte and J. A. Tenreiro Machado. A trajectory planning algorithm for redundant manipulators. In *Assembly and Task Planning, 1999.(ISATP'99) Proceedings of the 1999 IEEE International Symposium on*, pages 175–180, 1999.

[98] A. A. Maciejewski and C. A. Klein. Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments. *The international journal of robotics research*, 4(3):109–117, 1985.

[99] R. V. Patel, F. Shadpey, F. Ranjbaran, and J. Angeles. A collision-avoidance scheme for redundant manipulators: Theory and experiments. *Journal of Robotic Systems*, 22(12):737–757, 2005.

[100] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. In *Robotics and Automation. Proceedings. 1985 IEEE International Conference on*, volume 2, pages 500–505, 1985.

[101] N. A. Scott and C. R. Carignan. A line-based obstacle avoidance technique for dexterous manipulator operations. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 3353–3358, 2008.

[102] J. Canny. *The complexity of robot motion planning*. The MIT Press, 1988.

[103] C. C. Lin, L. W. Kuo, and J. H. Chuang. Potential-based path planning for robot manipulators. *Journal of Robotic Systems*, 22(6):313–322, 2005.

[104] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *The international journal of robotics research*, 5(1):90–98, 1986.

[105] J. O. Kim and P. K. Khosla. Real-time obstacle avoidance using harmonic potential functions. *Robotics and Automation, IEEE Transactions on*, 8(3):338–349, 1992.

[106] J. Baillieul. Avoiding obstacles and resolving kinematic redundancy. In *Robotics and Automation. Proceedings. 1986 IEEE International Conference on*, volume 3, pages 1698–1704, 1986.

[107] L. Sciavicco and B. Siciliano. A solution algorithm to the inverse kinematic problem for redundant manipulators. *Robotics and Automation, IEEE Journal of*, 4(4):403–410, 1988.

[108] F. T. Cheng, T. H. Chen, Y. S. Wang, and Y. Y. Sun. Obstacle avoidance for redundant manipulators using the compact QP method. In *Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on*, pages 262–269, 1993.

[109] Z. Y. Guo and T. C. Hsia. Joint trajectory generation for redundant robots in an environment with obstacles. *Journal of robotic systems*, 10(2):199–215, 1993.

[110] F. T. Cheng, T. H. Chen, and Y. Y. Sun. Resolving manipulator redundancy under inequality constraints. *Robotics and Automation, IEEE Transactions on*, 10(1):65–71, 1994.

[111] U. Sezgin, L. D. Seneviratne, and S. W. E. Earles. Redundancy utilization for obstacle avoidance of planar robot manipulators. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 211(6):463–475, 1997.

[112] O. Khatib. Dynamic control of manipulators in operational space. In *Sixth CISM-IFToMM Congress on Theory of Machines and Mechanisms*, pages 1128–1131, 1983.

[113] M. Vukobratovic and M. Kircanski. A dynamic approach to nominal trajectory synthesis for redundant manipulators. *IEEE transactions on systems, man, and cybernetics*, 14(4):580–586, 1984.

[114] K. Suh and J. Hollerbach. Local versus global torque optimization of redundant manipulators. In *Robotics and Automation. Proceedings. 1987 IEEE International Conference on*, volume 4, pages 619–624, 1987.

[115] A. Nedungadi and K. Kazerouinian. A local solution with global characteristics for the joint torque optimization of a redundant manipulator. *Journal of robotic systems*, 6(5):631–654, 1989.

[116] S. Ma, S. Hirose, and D. N. Nenchev. Improving local torque optimization techniques for redundant robotic mechanisms. *Journal of Robotic Systems*, 8(1):75–91, 1991.

[117] H. J. Kang and R. A. Freeman. Joint torque optimization of redundant manipulators via the null space damping method. In *Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on*, pages 520–525, 1992.

[118] W. J. Chung, W. K. Chung, and Y. Youm. Dynamic control of redundant manipulators using full row-rank minors of jacobian. In *Intelligent Robots and Systems' 93, IROS'93. Proceedings of the 1993 IEEE/RSJ International Conference on*, volume 2, pages 1053–1058, 1993.

[119] H. J. Kang and R. A. Freeman. Null space damping method for local joint torque optimization of redundant manipulators. *Journal of robotic systems*, 10(2):249–270, 1993.

[120] S. Ma and D. N. Nenchev. Local torque minimization for redundant manipulators: a correct formulation. *Robotica*, 14(02):235–239, 1996.

[121] S. Ma. A balancing technique to stabilize local torque optimization solution of redundant manipulators. *Journal of robotic systems*, 13(3):177–185, 1996.

[122] A. R. Hirakawa and A. Kawamura. Trajectory generation for redundant manipulators under optimization of consumed electrical energy. In *Industry Applications Conference, 1996. Thirty-First IAS Annual Meeting, IAS'96., Conference Record of the 1996 IEEE*, volume 3, pages 1626–1632, 1996.

[123] A. R. Hirakawa and A. Kawamura. Proposal of trajectory generation for redundant manipulators without inverse matrix calculation and application to consumed electrical energy minimization. *Advanced robotics*, 11(3):213–232, 1996.

[124] A. R. Hirakawa and A. Kawamura. Trajectory planning of redundant manipulators for minimum energy consumption without matrix inversion. In *Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on*, volume 3, pages 2415–2420, 1997.

[125] C. J. Wu. Minimum-torque trajectory planning of redundant manipulators between two joint configurations. *Journal of the Chinese Institute of Engineers*, 22(2):159–169, 1999.

[126] B. McAvoy, B. Sangolola, and Z. Szabad. Optimal trajectory generation for redundant planar manipulators. In *Systems, Man, and Cybernetics, 2000 IEEE International Conference on*, volume 5, pages 3241–3246, 2000.

[127] S. Lee and R. M. Kil. Redundant arm kinematic control with recurrent loop. *Neural Networks*, 7(4):643–659, 1994.

[128] G. Wu and J. Wang. A recurrent neural network for manipulator inverse kinematics computation. In *Neural Networks, 1994. IEEE World Congress on Computational Intelligence., 1994 IEEE International Conference on*, volume 5, pages 2715–2720, 1994.

[129] K. K. Aydin. Path planning and redundancy resolution for planar redundant manipulators, using artificial neural networks. In *Neural Networks, 1995. Proceedings., IEEE International Conference on*, volume 5, pages 2560–2565, 1995.

[130] H. Ding and S. P. Chan. Joint torque optimization for redundant manipulators using neural networks. In *Industrial Electronics, Control, and Instrumentation, 1995., Proceedings of the 1995 IEEE IECON 21st International Conference on*, volume 2, pages 1348–1353, 1995.

[131] Z. Mao and T. C. Hsia. Obstacle avoidance inverse kinematics solution of redundant robots by neural networks. *Robotica*, 15(1):3–10, 1997.

[132] W. S. Tang and J. Wang. Two recurrent neural networks for local joint torque optimization of kinematically redundant manipulators. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 30(1):120–128, 2000.

[133] C. Y. Chung and B. H. Lee. An approach to torque optimizing control for a redundant manipulator. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 4, pages 4066–4071, 2001.

[134] A. Khoukhi, K. Demirli, L. Baron, and M. Balazinski. Nuero-fuzzy multi-objective trajectory planning of redundant manipulators. In *Systems, Man and Cybernetics, 2007. ISIC. IEEE International Conference on*, pages 2831–2836, 2007.

[135] T. Arakawa, N. Kubota, and T. Fukuda. Virus-evolutionary genetic algorithm with subpopulations: application to trajectory generation of redundant manipulator through energy optimization. In *Systems, Man, and Cybernetics, 1996., IEEE International Conference on*, volume 3, pages 1930–1935, 1996.

[136] M. da Graca Marcos, J. A. Tenreiro Machado, and T. P. Azevedo-Perdicoulis. Trajectory planning of redundant manipulators using genetic algorithms. *Communications in Nonlinear Science and Numerical Simulation*, 14(7):2858–2869, 2009.

[137] J. C. W. Sullivan and A. G. Pipe. Path planning for redundant robot manipulators: a global optimization approach using evolutionary search. In *Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference on*, volume 3, pages 2396–2400, 1998.

[138] D. P. Martin, J. Baillieul, and J. M. Hollerbach. Resolution of kinematic redundancy using optimization techniques. *Robotics and Automation, IEEE Transactions on*, 5(4):529–533, 1989.

[139] S. Seereeram and J. T. Wen. A global approach to path planning for redundant manipulators. *Robotics and Automation, IEEE Transactions on*, 11(1):152–160, 1995.

[140] A. A. Ata and M. Y. Sa'adah. Soft motion trajectory for planar redundant manipulator. In *Control, Automation, Robotics and Vision, 2006. ICARCV'06. 9th International Conference on*, pages 1–6, 2006.

[141] J. Yang, J. Kim, E. P. Pitarch, and K. Abdel-Malek. Optimal trajectory planning for redundant manipulators based on minimum jerk. 2008.

[142] S. Ma and M. Watanabe. Minimum time path-tracking control of redundant manipulators. In *Intelligent Robots and Systems, 2000.(IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on*, volume 1, pages 27–32, 2000.

[143] A. A. Ata and T. R. Myo. Optimal point-to-point trajectory tracking of redundant manipulators using generalized pattern search. *Arxiv preprint cs/0601063*, 2006.

[144] E. Singla, S. Tripathi, V. Rakesh, and B. Dasgupta. Dimensional synthesis of kinematically redundant serial manipulators for cluttered environments. *Robotics and Autonomous Systems*, 58(5):585–595, 2010.

[145] K. K. Ayten, M. N. Sahinkaya, and P. Iravani. Optimum trajectory planning for redundant and hyper-redundant manipulators through inverse dynamics. *Pro-*

ceedings on International Design Engineering Technical Conferences & Computers and information in Engineering Conference, pages 1–8, 2011.

[146] T. Vittor, R. Willgoss, and T. Furukawa. Hyper-redundant manipulator control for reconfigurability and obstacle avoidance. In *Australian Conference on Robotics and Automation, Canberra*, 2004.

[147] S. Ma, S. Hirose, and H. Yoshinada. Development of a hyper-redundant multijoint manipulator for maintenance of nuclear reactors. *Advanced robotics*, 9(3):281–300, 1994.

[148] R. Buckingham and A. Graham. Nuclear snake-arm robots. *Industrial Robot: An International Journal*, 39(1):6–11, 2012.

[149] S. Yahya, M. Moghavvemi, S. S. Yang, and H. A. F. Mohamed. Motion planning of hyper redundant manipulators based on a new geometrical method. In *Industrial Technology, 2009. ICIT 2009. IEEE International Conference on*, pages 1–5, 2009.

[150] E. S. Conkur. Path following algorithm for highly redundant manipulators. *Robotics and Autonomous Systems*, 45(1):1–22, 2003.

[151] G. S. Chirikjian and J. W. Burdick. A hyper-redundant manipulator. *Robotics & Automation Magazine, IEEE*, 1(4):22–29, 1994.

[152] J. Gallardo-Alvarado, R. Lesso-Arroyo, and J. S. Garcia-Miranda. A worm-inspired new spatial hyper-redundant manipulator. *Robotica*, 29(4):571, 2011.

[153] G. S. Chirikjian and J. W. Burdick. An obstacle avoidance algorithm for hyper-redundant manipulators. In *Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on*, pages 625–631, 1990.

[154] S. Ma and M. Konno. An obstacle avoidance scheme for hyper-redundant manipulators-global motion planning in posture space. In *Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on*, volume 1, pages 161–166, 1997.

[155] C. C. Lin and J. H. Chuang. A potential-based path planning algorithm for hyper-redundant manipulators. *Journal of the Chinese Institute of Engineers*, 33(3):415–427, 2010.

[156] G. S. Chirikjian and J. W. Burdick. Kinematically optimal hyper-redundant manipulator configurations. *Robotics and Automation, IEEE Transactions on*, 11(6):794–806, 1995.

[157] F. B. M. Duarte and J. A. Tenreiro Machado. Kinematic optimization of redundant and hyper-redundant robot trajectories. In *Electronics, Circuits and Systems, 1998 IEEE International Conference on*, volume 2, pages 467–470, 1998.

[158] M. P. J. Fromherz and W. Jackson. Predictable motion of hyper-redundant manipulators using constrained optimization control. In *Int. Conf. on AI 2000*, pages 1141–1148, 2000.

[159] U. G. Matlab. The MathWorks inc. *Natick, MA*, 2003.

[160] M. G. Marcos, J. A. T. Machado, and T. P. Azevedo-Perdicoulis. An evolutionary approach for the motion planning of redundant and hyper-redundant manipulators. *Nonlinear Dynamics*, 60(1):115–129, 2010.

[161] P. S. Donelan. Singularities of robot manipulators. *Singularity Theory*, pages 189–217, 2007.